Practical Electronics

The UK's premier electronics and computing maker magazine

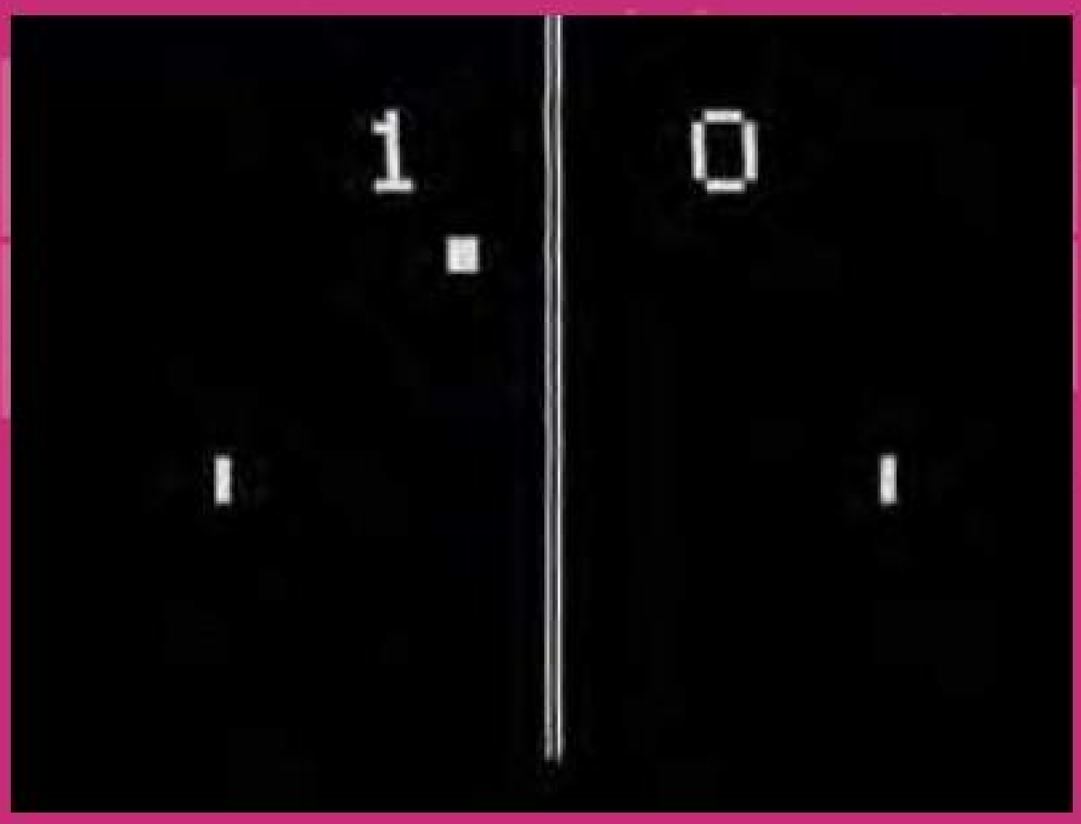
Circuit Surgery
Using distortion and distortion circuits

KickStart
Exploring DACs and microcontrollers

Make it with Micromite
MMBASIC + RPi Pico + display = PicoMite Backpack!

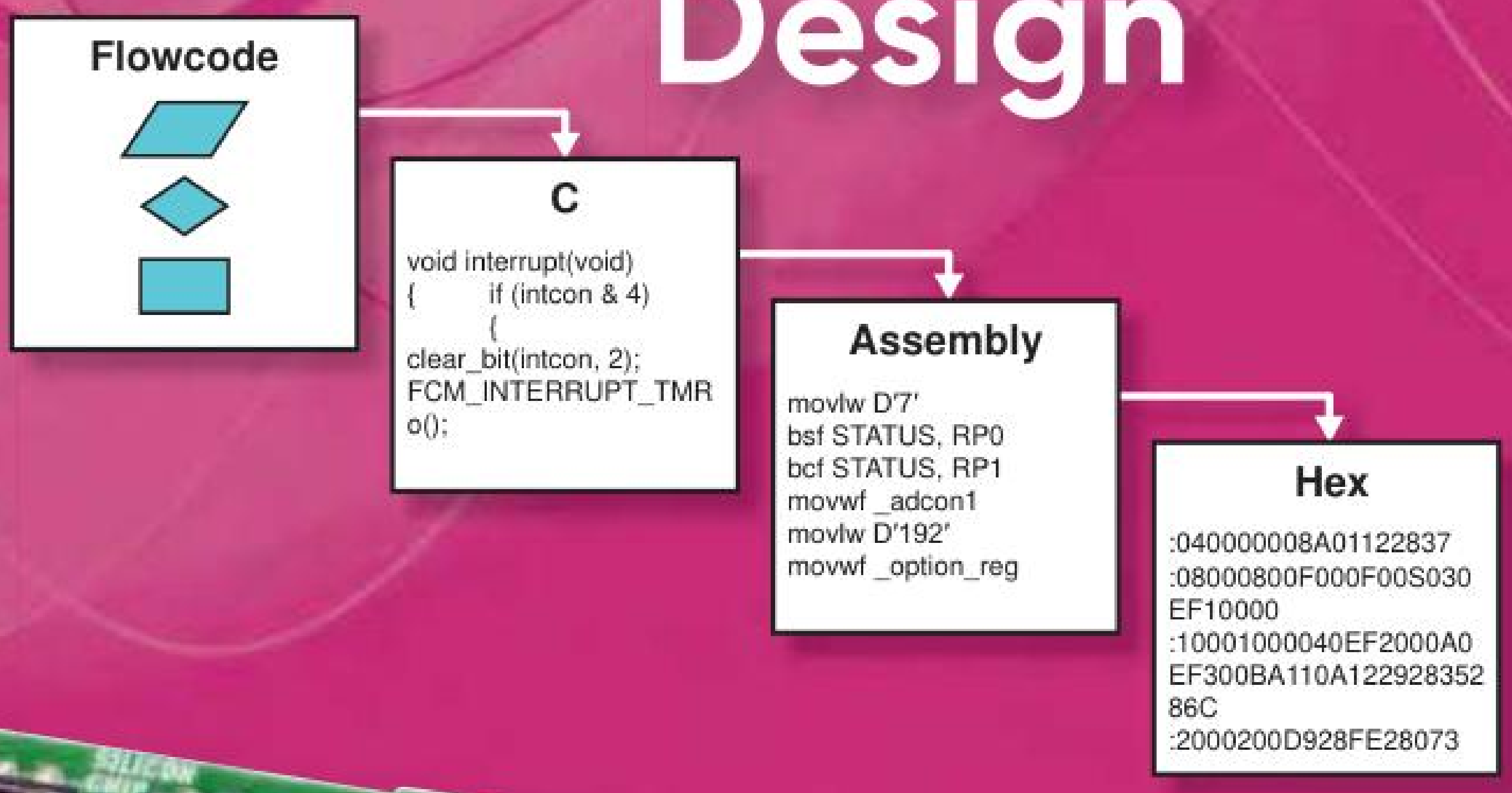
Multi-purpose Battery Manager

WIN!
Microchip SAM V71 Xplained Ultra Evaluation Kit
WIN!



Retro gaming with Nano Pong!

Flowcode Digital Clock Design



C

```
void interrupt(void)
{         if (intcon & 4)
          {
clear_bit(intcon, 2);
FCM_INTERRUPT_TMR
o();
```

Assembly

```
movlw D'7'
bsf STATUS, RP0
bcf STATUS, RP1
movwf _adcon1
movlw D'192'
movwf _option_reg
```

Hex

```
:040000008A01122837
:08000800F000F00S030
EF10000
:10001000040EF2000A0
EF300BA110A122928352
86C
:2000200D928FE28073
```

Simple MIDI Music Keyboard

PLUS!

Techno Talk – Time for a total rethink?
Cool Beans – Touch-sensitive robots and using servos
Net Work – The irresistible rise of automotive electronics

Aug 2022 £5.49

www.electronpublishing.com | @practicalelec | practicalelectronics

Put the “Smart” in Smart Home

Easy Design with the PIC16F18076 Family of MCUs

No matter what you’re building, the PIC16F18076 family of MCUs can help you create faster, more feature-rich smart home technology. Now, you can easily add capacitive touch buttons, control motors, integrate IoT connectivity and monitor system health using the family’s on-chip smart peripherals. And, these cost-effective MCUs integrate seamlessly with our latest GUI-based code configuration tools for a refreshingly simple development experience. Start building your next smart home appliance today using Curiosity Nano rapid prototyping hardware.

Key Features

- 8 to 44-pin packages
- 3.5 KB to 28 KB of internal memory
- CPU Speed up to 32 MHz
- 10-bit Analog-to-Digital Converter with Computation (ADCC)
- Automated Capacitive Voltage Divider (CVD)
- 8-bit Digital-to-Analog Converter (DAC) module

MICROCHIP
PIC16F18076

microchip.com/PIC16F180

The Microchip name and logo and the Microchip logo
are registered trademarks of Microchip Technology
Incorporated in the U.S.A. and other countries. All other
trademarks are the property of their registered owners.
© 2022 Microchip Technology Inc. All rights reserved.
MEC2423A-UK-05-22

Practical
Electronics

Volume 51. No. 8
August 2022
ISSN 2632 573X

Contents

Projects and Circuits

Multi-purpose Battery Manager by Tim Blythman 14
Building on our recent Battery Multi Logger, this design not only monitors what's going on but also connects and disconnects loads based on battery charge state.

Simple MIDI Music Keyboard by Tim Blythman 26
This MIDI Keyboard is a follow-up to our 64-key MIDI Matrix. It is just as flexible and offers an easy way to make music, plus it can be repurposed for many other uses.

Nano Pong by Tim Blythman 30
Our Nano Pong game is modern and retro at the same time; it replaces the 70-odd discrete logic chips in the original with a single chip that costs about £1!

USB Power Delivery Chargers by Jim Rowe 36
Learn about low-cost modules that have appeared recently to take advantage of the dramatic growth in USB capability, especially in the area of power delivery (PD).

Series, Features and Columns

Techno Talk by Mark Nelson 8
Time for a total rethink?

Net Work by Alan Winstanley 10
A round-up of topical technological issues and trends from the UK and around the world.

Make it with Micromite by Phil Boyce 40
Part 38: How to build a PicoMite BackPack

KickStart by Mike Tooley 47
Part 9: Exploring microcontroller digital-to-analogue conversion

Max's Cool Beans by Max The Magnificent 54
Flashing LEDs and drooling engineers – Part 30

Flowcode Graphical Programming by Martin Whitlock 58
Part 5: Building a Digital Clock

Circuit Surgery by Ian Bell 65
Distortion and distortion circuits – Part 3

Regulars and Services

Wireless for the Warrior 2

Subscribe to Practical Electronics and save money 4

Reader services – Editorial and Advertising Departments 7

Editorial 7
The power of batteries

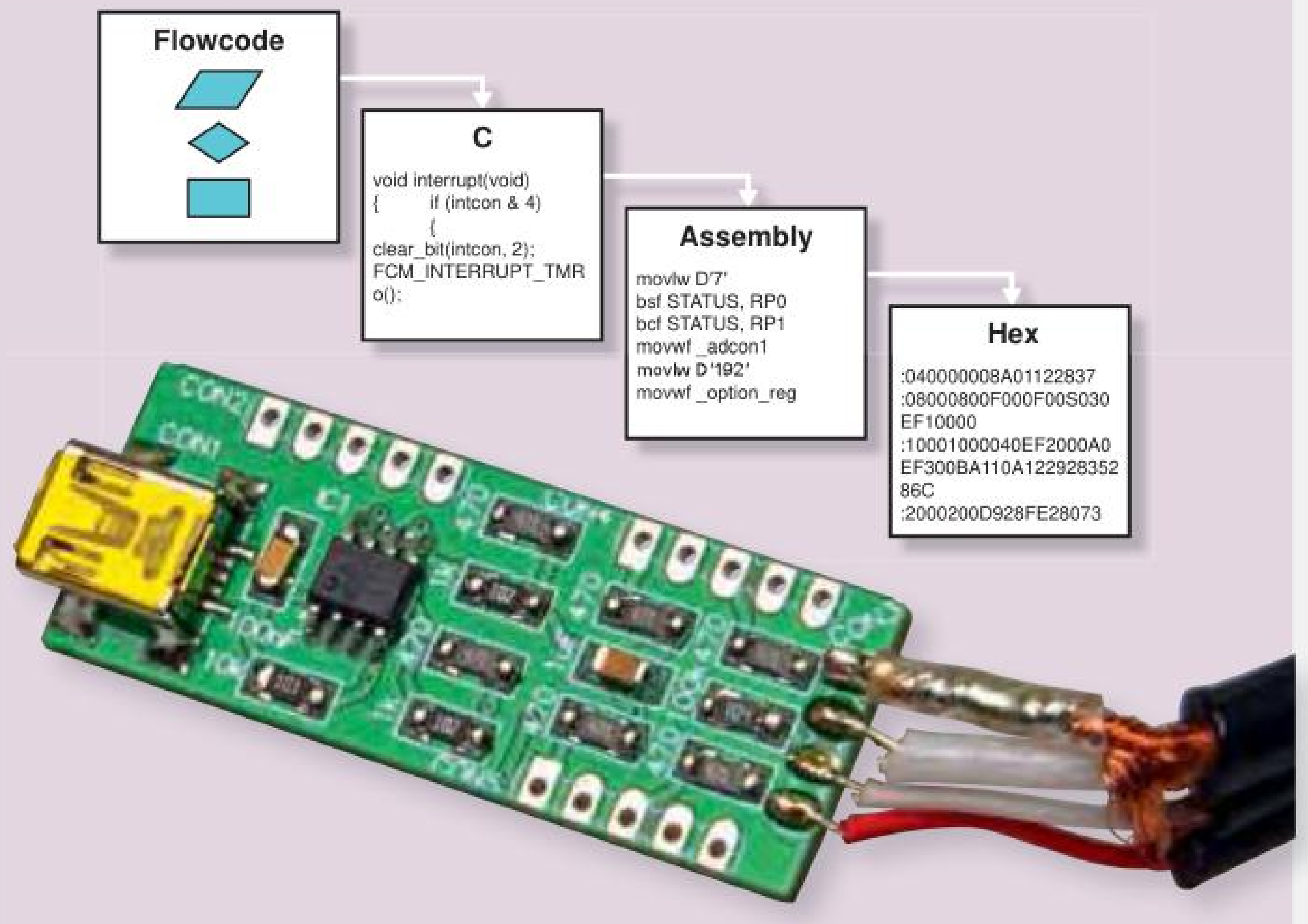
Exclusive Microchip reader offer 9
Win a Microchip SAM V71 Xplained Ultra Evaluation Kit

PE Teach-In 8 25

Practical Electronics PCB Service 70
PCBs for Practical Electronics projects

Classified ads and Advertiser index 71

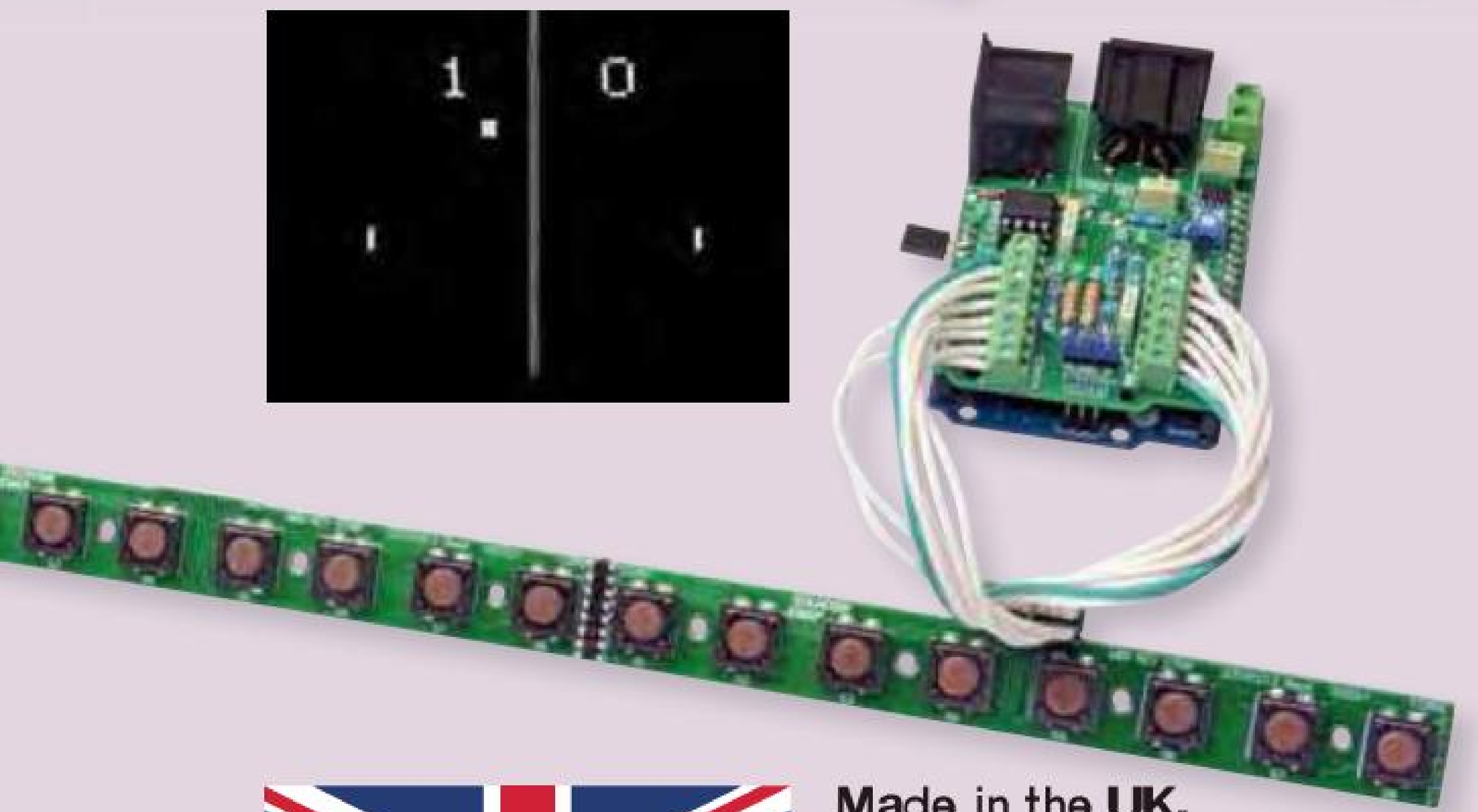
Next month! – highlights of our next issue of Practical Electronics 72



```
void interrupt(void)
{         if (intcon & 4)
          {
clear_bit(intcon, 2);
FCM_INTERRUPT_TMR
o():
```

```
movlw D'7'
bsf STATUS, RP0
bcf STATUS, RP1
movwf _adcon1
movlw D'192'
movwf _option_reg
```

```
:040000008A01122837
:08000800F000F00S030
EF10000
:10001000040EF2000A0
EF300BA110A122928352
86C
:2000200D928FE28073
```



Made in the UK.

Written in Britain, Australia, the US and Ireland.

Read everywhere.

© Electron Publishing Limited 2022

Copyright in all drawings, photographs, articles, technical designs, software and intellectual property published in *Practical Electronics* is fully protected, and reproduction or imitation in whole or in part are expressly forbidden.

The September 2022 issue of *Practical Electronics* will be published on Thursday, 4 August 2022 – see page 72.

WIRELESS FOR THE WARRIOR

by LOUIS MEULSTEE

THE DEFINITIVE TECHNICAL HISTORY OF RADIO COMMUNICATION EQUIPMENT IN THE BRITISH ARMY

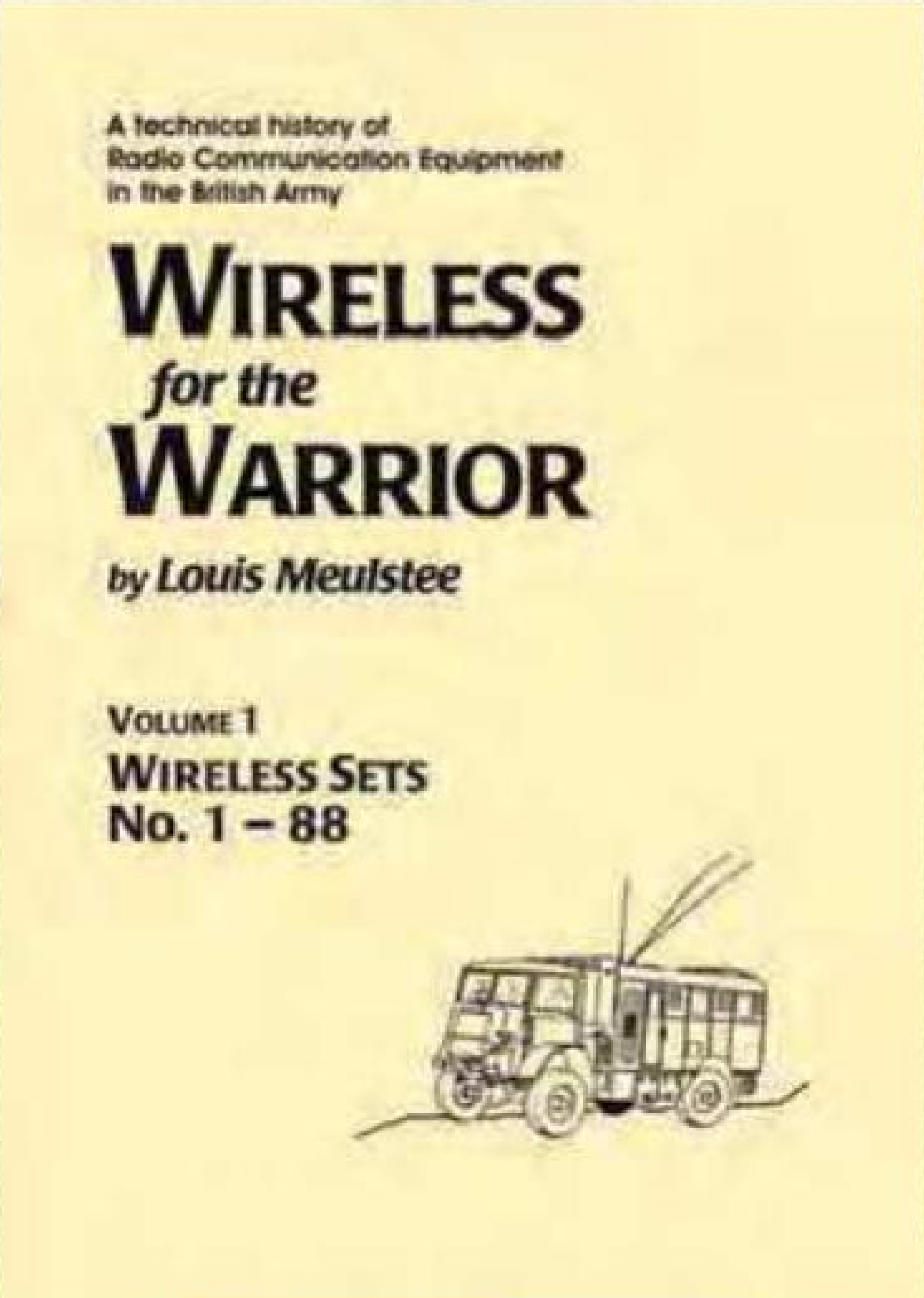
The *Wireless for the Warrior* books are a source of reference for the history and development of radio communication equipment used by the British Army from the very early days of wireless up to the 1960s.

The books are very detailed and include circuit diagrams, technical specifications and alignment data, technical development history, complete station lists and vehicle fitting instructions.

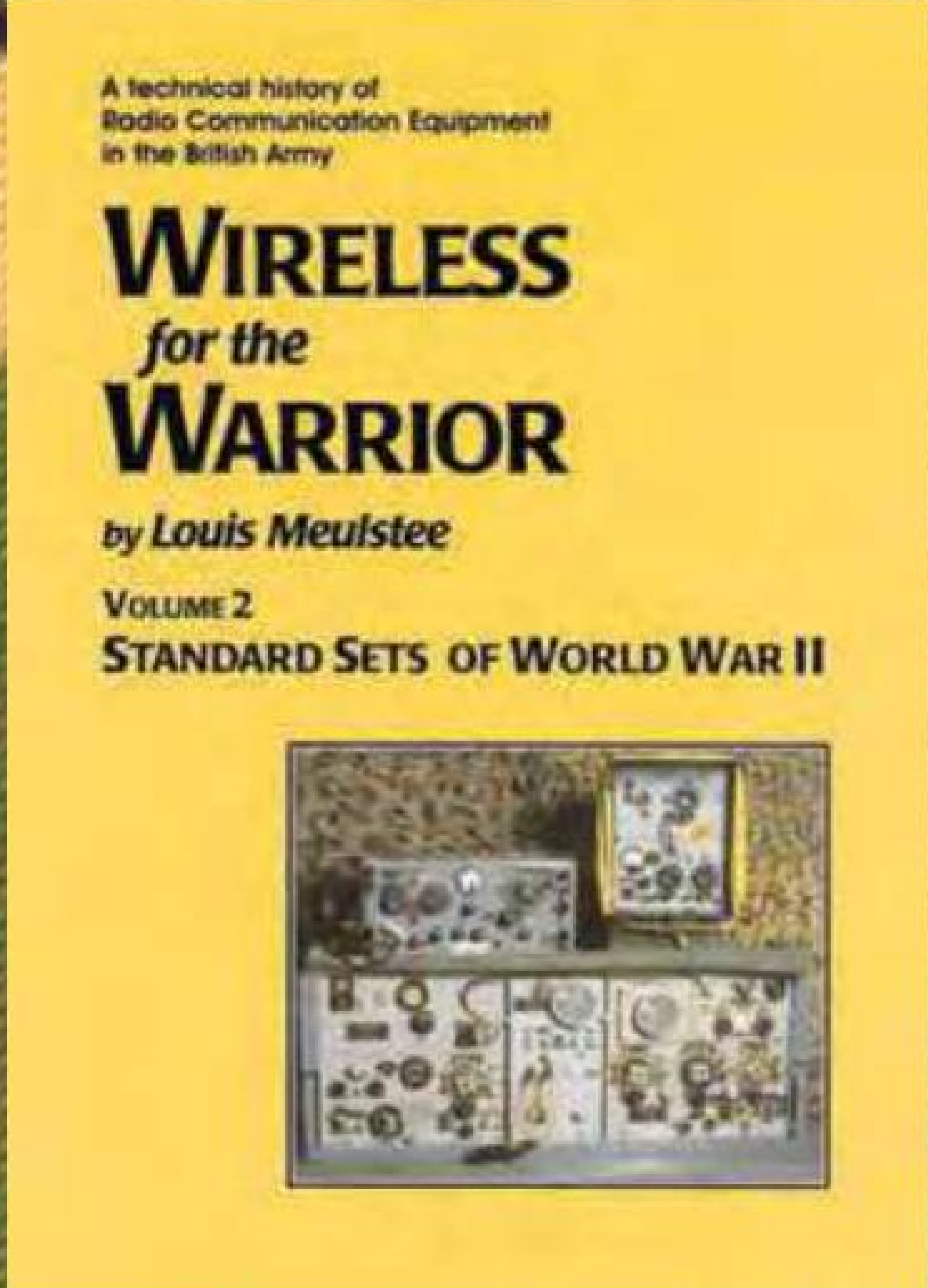
Volume 1 and *Volume 2* cover transmitters and transceivers used between 1932-1948. An era that starts with positive steps taken to formulate and develop a new series of wireless sets that offered great improvements over obsolete World War I pattern equipment. The other end of this timeframe saw the introduction of VHF FM and hermetically sealed equipment.

Volume 3 covers army receivers from 1932 to the late 1960s. The book not only describes receivers specifically designed for the British Army, but also the Royal Navy and RAF. Also covered: special receivers, direction finding receivers, Canadian and Australian Army receivers, commercial receivers adopted by the Army, and Army Welfare broadcast receivers.

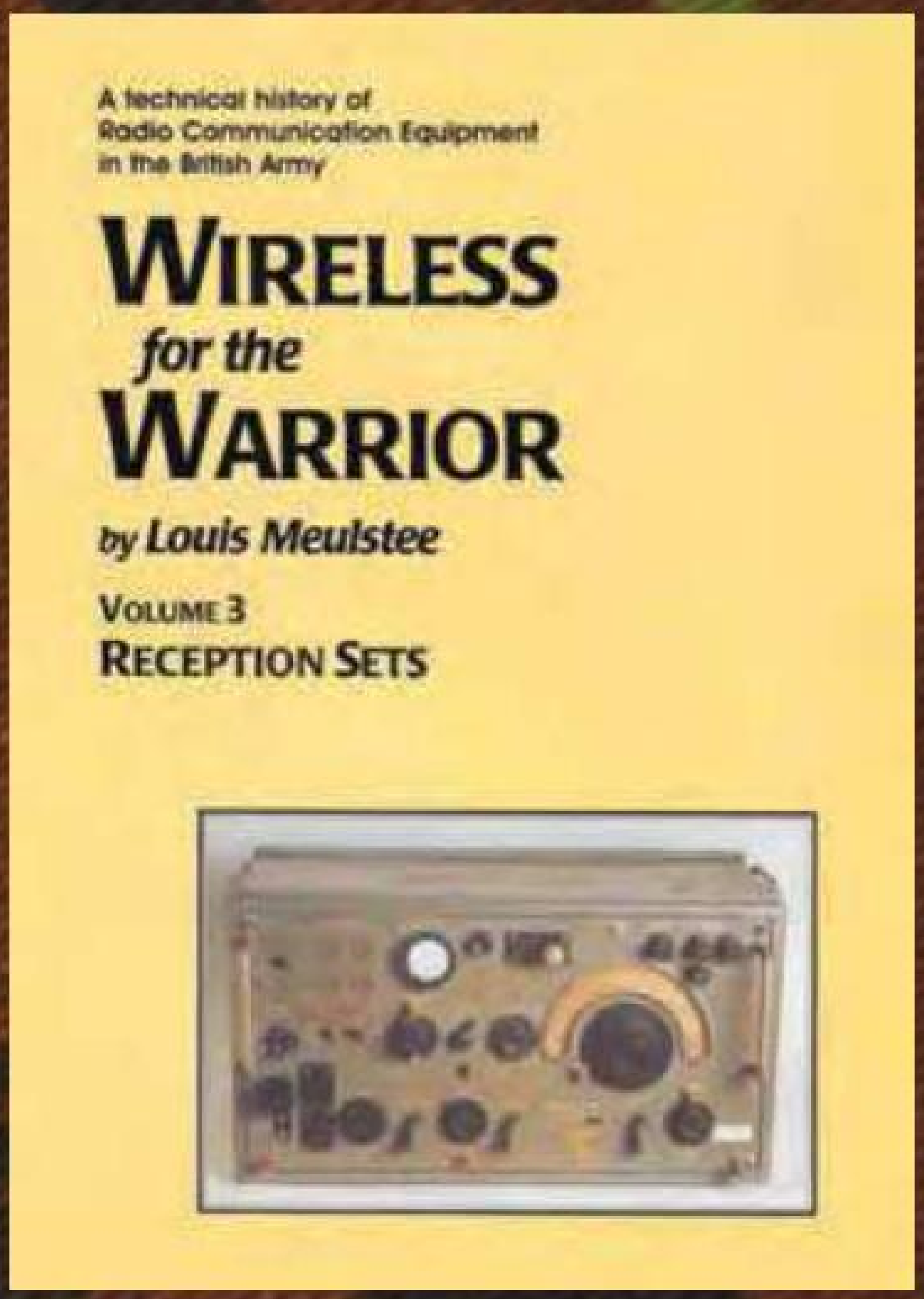
Volume 4 covers clandestine, agent or 'spy' radio equipment, sets which were used by special forces, partisans, resistance, 'stay behind' organisations, Australian Coast Watchers and the diplomatic service. Plus, selected associated power sources, RDF and intercept receivers, bugs and radar beacons.



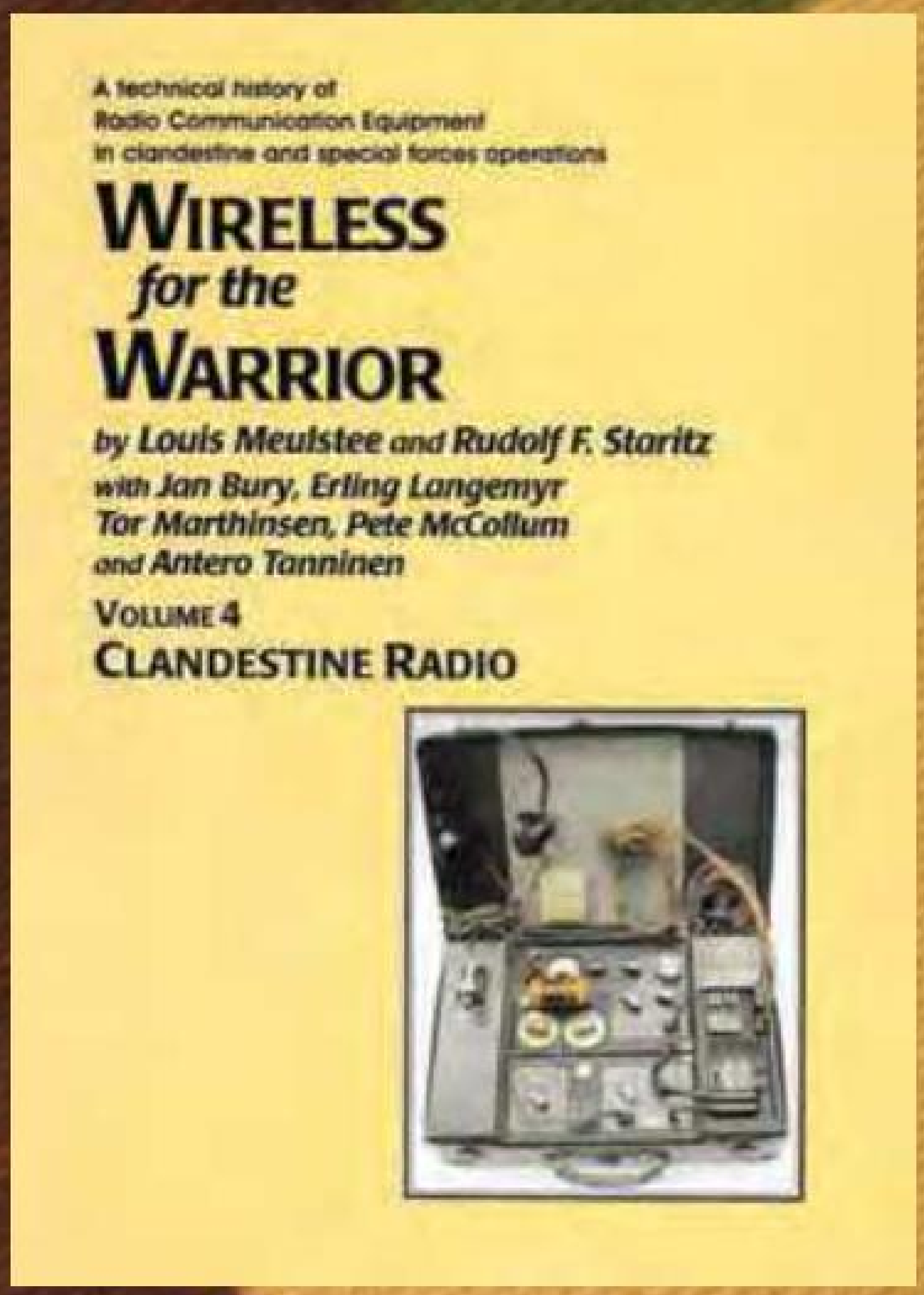
A technical history of
Radio Communication Equipment
in the British Army
WIRELESS
for the
WARRIOR
by Louis Meulstee
VOLUME 1
WIRELESS SETS
No. 1 – 88



A technical history of
Radio Communication Equipment
in the British Army
WIRELESS
for the
WARRIOR
by Louis Meulstee
VOLUME 2
STANDARD SETS OF WORLD WAR II



A technical history of
Radio Communication Equipment
in the British Army
WIRELESS
for the
WARRIOR
by Louis Meulstee
VOLUME 3
RECEPTION SETS



A technical history of
Radio Communication Equipment
in clandestine and special forces operations
WIRELESS
for the
WARRIOR
by Louis Meulstee and Rudolf F. Staritz
with Jan Bury, Erling Langemyr
Tor Marthinsen, Pete McCollum
and Antero Tanninen
VOLUME 4
CLANDESTINE RADIO

ORDER YOURS TODAY!
JUST CALL 01202 880299 OR VISIT www.electronpublishing.com

Quasar Electronics Limited
PO Box 6935, Bishops Stortford
CM23 4WP, United Kingdom

Tel: 01279 467799
E-mail: sales@quasarelectronics.co.uk
Web: quasarelectronics.co.uk

All prices include 20% VAT. Free UK mainland delivery on orders over £60. Postage & Packing Options (Up to 1Kg gross weight): **UK Standard 2-5 Day Delivery -** £4.95 : **UK Mainland Next Day Delivery -** £9.95 : Please order online if you reside outside the UK (our website will calculate postage for you).
Payment: We accept all major credit/debit cards. Make UK cheques/PO's payable to Quasar Electronics Limited and include P&P detailed above.
!! Order online for reduced price postage and fast despatch !!
Please visit our online shop now for full details of over 1000 electronic kits, projects, modules and publications. Discounts for bulk quantities.

QuasarElectronics.co.uk

Solutions for Home, Education & Industry Since 1993

Official Main Dealer stocking the full range of Kits, Modules, Robots, Instruments, Tools and much, much more...

Electronic Kits & Modules

We have a massive selection of self-assembly electronic kits and pre-assembled modules. Please see the full range on our website or call for details.

LED Buddy / LED Tester Kit

Hold any type of LED to the contact pads to see it's polarity, forward voltage & the recommended series resistor value. Adjustable target current & forward voltage. Great design aid.
*Order Code: WSMI198 - **£16.66***

5-in-1 Mechanical Coding Robot Kit

Mission-based, entry level STEM (Science - Technology - Engineering - Mathematics) coding robot designed to inspire users to learn about mechanical engineering & coding basics. Solderless assembly. 2x AAA battery powered (not included). Aged 14+.
Order Code: KSR19 - ~~£32.95~~ £29.95

Stereo Ultrasonic Bat Detector Kit

This exciting, educational & fun kit converts high frequency sounds (20 - 90kHz) normally imperceptible to humans like bat signals into audible noise. Can also help detect failures in machines, engines, etc. Stereo feature adds the possibility to pinpoint the source. Frequency range 20-90kHz. 3.5mm jack output. 129x60x40mm. Requires 3x AA batteries and stereo headphones (not included).
*Order Code: K8118 - **£20.39***

Bidirectional DC Motor Speed Controller

Control the speed of most common DC motors (rated up to 32Vdc/5A) in both the forward and reverse directions. The range of control is from fully OFF to fully ON in both directions. The direction and speed are controlled using a single potentiometer. Screw terminal block for connections. PCB: 90x42mm. **Not suitable for use with lead acid batteries!**
*Kit Order Code: 3166KT - **£19.96***
*Assembled Order Code: AS3166 - **£29.95***

3 Channel RGB LED Light Organ Kit

Sens
Low R
Mid G
High B
velleman R-G-B LED light organ

3 outputs react to different sound frequencies. On-board microphone picks up surrounding sound or music and drives the low, mid & high frequency outputs. Connect a 12-24Vdc RGB LED strip or separate colour LED strips or LEDs (not included). Master & separate channel sensitivity adjustment. 1.25 A max. per channel. Panel mountable facia 127 x 44mm. Terminal block or jack power supply connection. Control knobs included.
*Order Code: WSL209 - **£14.10***

Stereo Preamplifier with Tone Controls

Hi-fi quality stereo preamplifier board with very pleasing performance. Volume control with integrated power switch. +/-15 dB Bass and Treble tone controls. Channel balance control. RCA jacks f or audio inputs. Up to 20 dB of voltage gain. Selectable loudness function. Philips TDA1524A based circuit.
*Kit Order Code: 3100KT - **£23.95***
*Assembled Order Code: AS3100 - **£32.95***

The Lady Soldering Kit

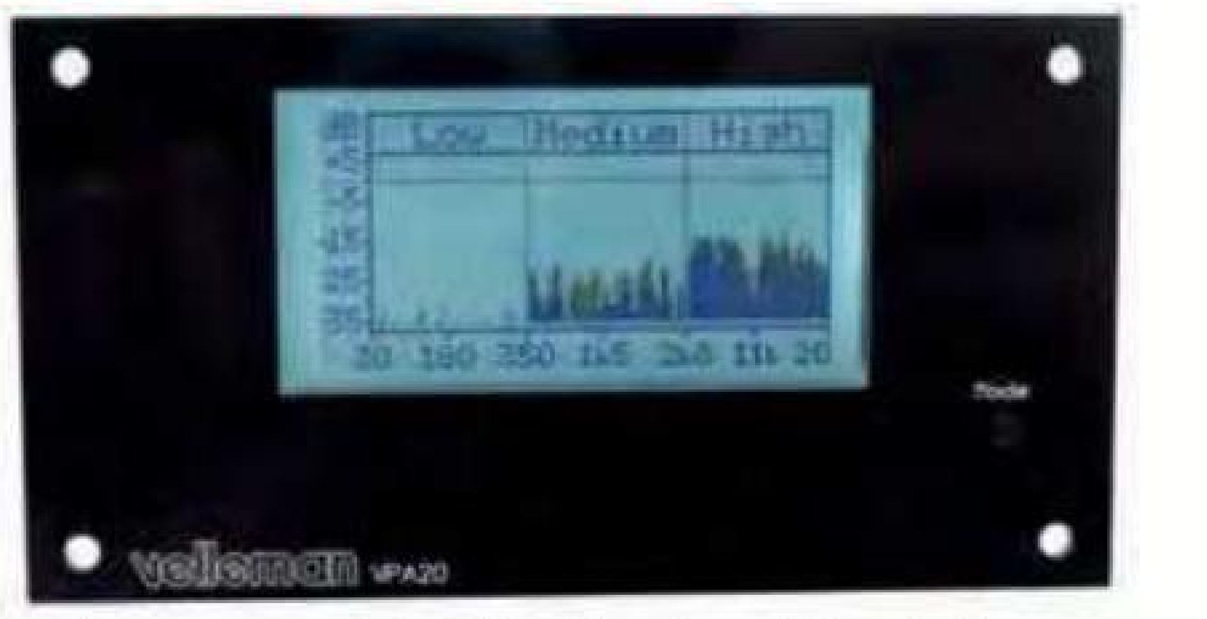
Meet the Lady! This self-assembly soldering kit includes both through-hole and SMD components, LEDs plus a beautiful red and white PCB.
*Order Code: WSAK206 - **£8.99***

Card Sales & Enquiries 01279 467799

Digitally Controlled FM Radio Kit

Build your own modern, high quality FM receiver project with excellent sensitivity powered by a simple 9V PP3 battery (not included. Auto-seeking button. 4 station presets. Volume control. Excellent learning project for schools and colleges.
*Order Code: MK194N - **£20.39***

Audio Analyser Display Kit

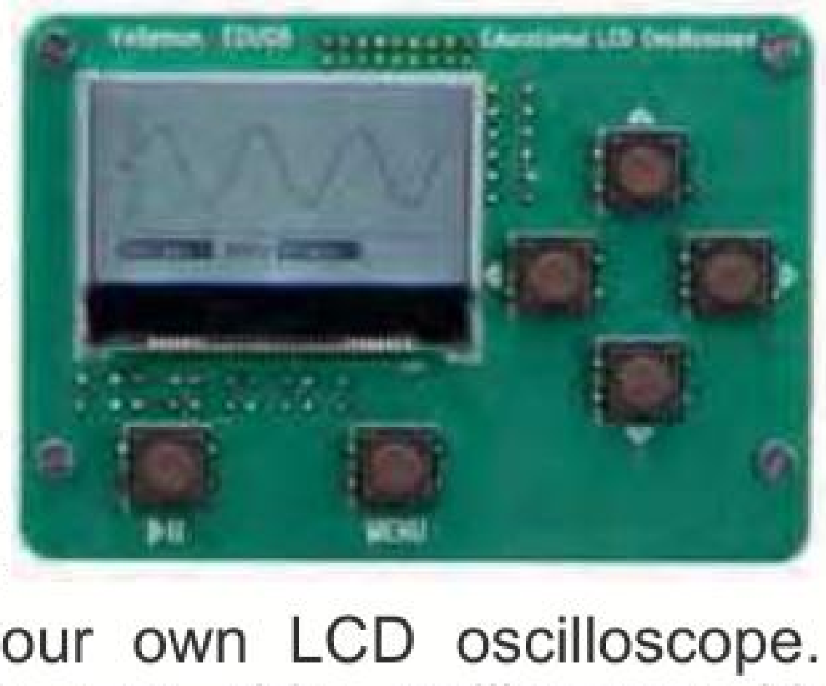


Small, compact LCD display, ideal for panel mounting. Give your homemade audio gear a high-tech look. Upgrade existing equipment. Provides Peak Power, RMS Power, Mean dB, Peak dB, Linear Audio Spectrum And 1/3 Octave Audio Spectrum. Auto / Manual range selection. Peak-hold function. Speaker impedance selection.
*Order Code: K8098 - **£38.39***

Electronic Component Tester Kit

Build your own versatile component tester. Shows value and pin layout information for resistors (0.1 Ohm resolution, max. 50 MOhm), coils (0.01mH - 20H), capacitors (28pF - 100mF), diodes, BJT, JFET, E-IGBT, D-IGBT, E-MOS & D-MOS.
*Order Code: K8115 - **£44.15***

LCD Oscilloscope Educational Kit



Build your own LCD oscilloscope with this exciting new kit. Learn how to read signals. See the electronic signals you learn about displayed on your own LCD oscilloscope. Despite the low cost, this oscilloscope kit has a lot of features found on expensive units like signal markers, frequency, dB, true RMS readouts and more. A powerful auto-setup function will get you going in a flash!
*Order Code: EDU08 - **£48.54***

Practical Electronics

UK readers SAVE £1 on every issue

SUBSCRIBE NOW!
01202 087631

Take out a one-year subscription and save more than £10 over the year.
Even better – save over £1 per issue if you subscribe for two years – a total saving of £26.

Overseas rates represent exceptional value

You also:
- **Avoid any cover price increase for the duration of your subscription**
- **Get your magazine delivered to your door each month**
- **Guarantee a copy, even if your local newsagent sells out**

Order by:
- **Phone or post with a cheque, postal order or credit card**
- **Or order online at: www.electronpublishing.com**

Practical Electronics

SUBSCRIPTION PRICES

Subscriptions for delivery direct to any address in:

UK: 6-months £29.99, 12-months £54.99, 24-months £104.99

Europe Airmail: 6-months £33.99, 12-months £63.99, 24-months £119.99

Rest Of The World Airmail: 6-months £41.99, 12-months £77.99, 24-months £149.99

Cheques (in **£ sterling only**) payable to *Practical Electronics* and sent to: Practical Electronics Subscriptions, PO Box 6337, Bournemouth BH1 9EH, United Kingdom

Tel: 01202 087631
Email: pesubs@selectps.com

Also via our online shop at**: www.electronpublishing.com**

Subscriptions start with the next available issue. We accept MasterCard or Visa.
(For past issues see the Back Issues page.)

DIGITAL SUBSCRIPTIONS

Electronic subscriptions are available from £24.99 for 12 months, more details at: **www.electronpublishing.com**

SUBSCRIPTION ORDER FORM

☐ **6 Months: UK £29.99, Europe £33.99 (Airmail), Rest of the World £41.99 (Airmail)**
☐ **1 Year: UK £54.99, Europe £63.99 (Airmail), Rest of the World £77.99 (Airmail)**
☐ **2 Years: UK £104.99, Europe £119.99 (Airmail), Rest of the World £149.99 (Airmail)**

To: **Practical Electronics Subscriptions**
PO Box 6337
Bournemouth BH1 9EH
United Kingdom

Tel: 01202 087631
Email: pesubs@selectps.com

I enclose payment of £ ....
(cheque/PO in £ sterling only)
payable to *Practical Electronics*

☐ Please charge my Visa/Mastercard

My card number is: ....
Please print clearly, and check that you have the number correct

Card Security Code .... Valid From Date........................
(The last 3 digits on or just under the signature strip)

Card Ex. Date ....

Name ....

Address ....

Post code .... Tel. ....

Email ....

Subscriptions start with the next available issue.

Subscribe to SILICON CHIP

SILICON CHIP
DECEMBER 2021
The small but powerful Hummingbird Amplifier
Hands-on with the Raspberry Pi Pico
The smallest Raspberry Pi yet!

Australia's top electronics magazine

Silicon Chip is one of the best DIY electronics magazines in the world. Each month is filled with a variety of projects that you can build yourself, along with features on a wide range of topics from in-depth electronics articles to general tech overviews.

	Print	Combined	Online
6 months	\$100 ≈£54 ≈€64	\$110 ≈£60 ≈€70	\$50 ≈£27 ≈€31
1 year	\$195 ≈£105 ≈€124	\$215 ≈£116 ≈€137	\$95 ≈£51 ≈€60
2 years	\$380 ≈£205 ≈€242	\$415 ≈£224 ≈€264	\$185 ≈£100 ≈€118

- *Combined subscriptions include both the printed magazine and online access.*
- *Prices are in Australian dollars (AUD) unless otherwise specified.*
- *Prices listed are for everywhere in the world except Australia and New Zealand. View siliconchip.com.au/Shop/SubRates for a full list of current prices.*
- *Price estimates are just that and should only be used as a guide, we primarily take payment in the form of Australian dollars.*

Try our Online Subscription – now with PDF downloads!

An online issue is perfect for those who don't want too much clutter around the house and is the same price worldwide. Issues can be viewed online, or downloaded as a PDF.

Published in Silicon Chip

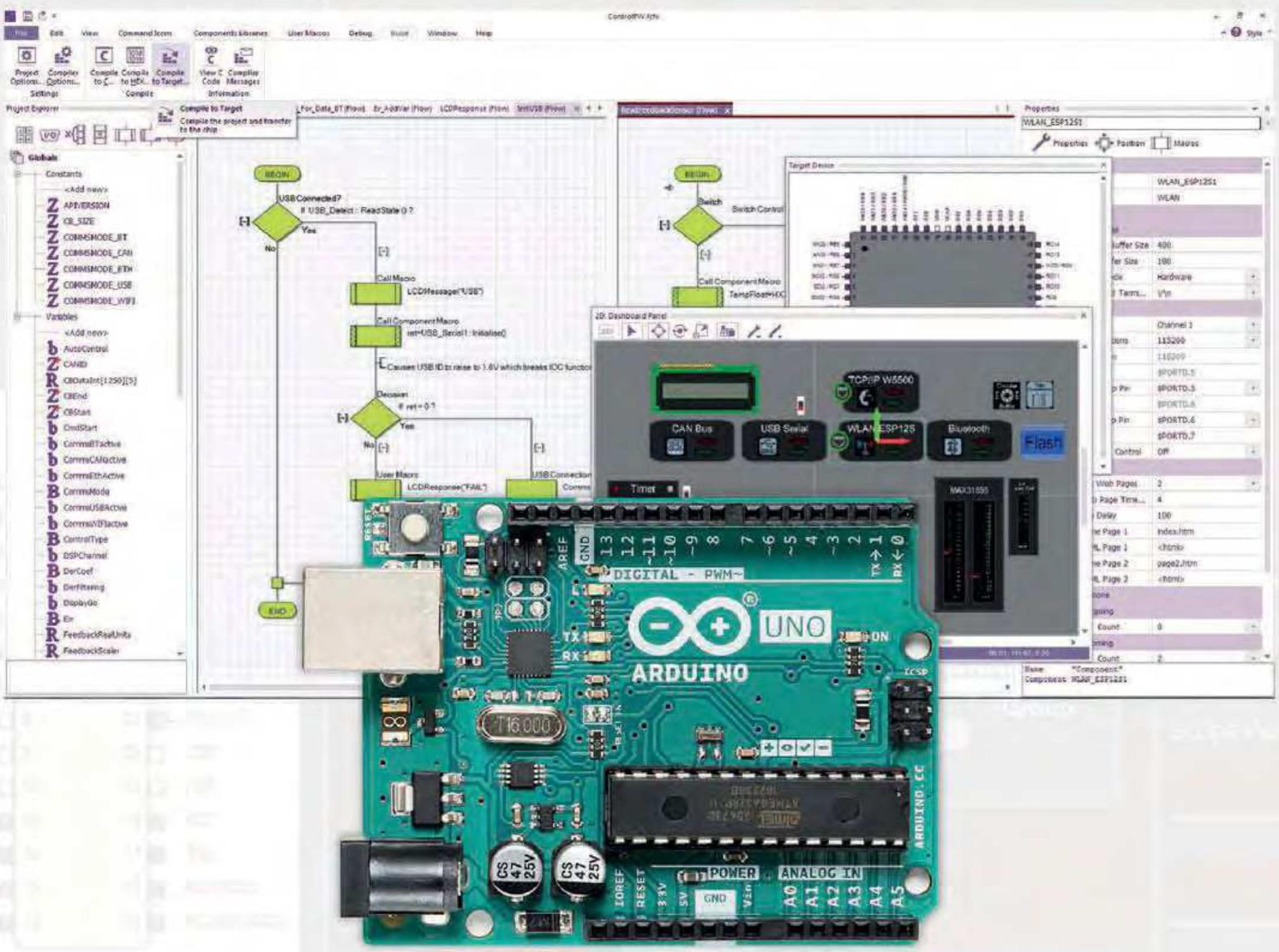
Big Brother is Tracking You; November 2021

USB Cable Tester; November 2021

Hummingbird Amp; December 2021

Raspberry Pi Pico; December 2021

To start your subscription go to siliconchip.com.au/Shop/Subscribe


BEGIN
USB Connected?
Call Macro
LCDMessage("USB")
Call Component Macro
ret=USB_Serial1::Initialise()
Decision
User Macro
LCDResponse("FAIL")
END
2D Dashboard Panel
CAN Bus
USB Serial
TCP/IP W5500
WLAN ESP12S
Bluetooth
Flash
Timer
MAX31855
ARDUINO
UNO
DIGITAL - PWM~
POWER
ANALOG IN
ARDUINO.CC

CREATE FANTASTIC ELECTRONIC SYSTEMS USING

FLOWCODE

20% off your Flowcode purchase using code: **EPE20**

Use code at checkout: flowcode.co.uk/buy

Practical Electronics

Volume 51. No. 8
August 2022
ISSN 2632 573X

Editorial

Editorial offices
Practical Electronics
Electron Publishing Limited
1 Buckingham Road
Brighton
East Sussex BN1 3RA
Tel 01273 777619
Mob 07973 518682
Fax 01202 843233
Email pe@electronpublishing.com
Web www.electronpublishing.com

Advertisement offices
Practical Electronics Adverts
1 Buckingham Road
Brighton
East Sussex BN1 3RA
Tel 01273 777619
Mob 07973 518682
Email pe@electronpublishing.com

Editor Matt Pulzer
General Manager Louisa Pulzer
Digital subscriptions Stewart Kearn **Tel** 01202 880299
Online Editor Alan Winstanley
Web Systems Kris Thain
Publisher Matt Pulzer

Print subscriptions
Practical Electronics Subscriptions
PO Box 6337
Bournemouth BH1 9EH
United Kingdom
Tel 01202 087631
Email pesubs@selectps.com

Technical enquiries
We regret technical enquiries cannot be answered over the telephone. We are unable to offer any advice on the use, purchase, repair or modification of commercial equipment or the incorporation or modification of designs published in the magazine. We cannot provide data or answer queries on articles or projects that are more than five years old.

Questions about articles or projects should be sent to the editor by email: pe@electronpublishing.com

Projects and circuits
All reasonable precautions are taken to ensure that the advice and data given to readers is reliable. We cannot, however, guarantee it and we cannot accept legal responsibility for it.

A number of projects and circuits published in *Practical Electronics* employ voltages that can be lethal. You should not build, test, modify or renovate any item of mains-powered equipment unless you fully understand the safety aspects involved and you use an RCD (GFCI) adaptor.

Component supplies
We do not supply electronic components or kits for building the projects featured, these can be supplied by advertisers. We advise readers to check that all parts are still available before commencing any project in a back-dated issue.

Advertisements
Although the proprietors and staff of *Practical Electronics* take reasonable precautions to protect the interests of readers by ensuring as far as practicable that advertisements are bona fide, the magazine and its publishers cannot give any undertakings in respect of statements or claims made by advertisers, whether these advertisements are printed as part of the magazine, or in inserts. The Publishers regret that under no circumstances will the magazine accept liability for non-receipt of goods ordered, or for late delivery, or for faults in manufacture.

Transmitters/bugs/telephone equipment
We advise readers that certain items of radio transmitting and telephone equipment which may be advertised in our pages cannot be legally used in the UK. Readers should check the law before buying any transmitting or telephone equipment, as a fine, confiscation of equipment and/or imprisonment can result from illegal use or ownership. The laws vary from country to country; readers should check local laws.

The power of batteries

We have a great set of projects for you this month. An easy-to-build *MIDI keyboard*; *Nano Pong*, a modern twist on a retro game; and a *PicoMite Backpack* that combines the power of MMBasic, a high-quality touchscreen display and an inexpensive but very capable Raspberry Pi Pico microcontroller. But that's not all – my favourite in this issue is a very useful *Battery Manager* that combines three recent projects (*Battery Multi-logger*, *Battery Balance*r, and a *Mini Isolated Serial Link*) with some new circuits to produce a handy design that is compact, flexible and modular. It can connect and disconnect up to four loads/sources to protect batteries, and is capable of switching well over 20A at 10V-60V. If you're in the market for battery charging designs that go above and beyond basic 'electron pumping' then this is one for you.

Back in the early days of *PE*, most battery circuits really only meant one thing, some variation on a simple lead-acid battery charger. They were relatively crude, mostly safe and worked reasonably well, but there wasn't a lot of sophistication built into the designs.

Fast forward to 2022 and the range of battery technologies, cell designs and supporting circuity is huge. Power density and charge time have improved in leaps and bounds, but so too has the need to charge/discharge properly – get that wrong and at best you will damage a cell, at worst it may catch fire. Lithium-based batteries have become the cornerstone of many technologies that run untethered from the mains, from in-ear headphones to electric vehicles, but they are very picky about how you use them and do not tolerate abuse.

As the world moves away from fossil fuels and towards distributed generation systems (photovoltaic panels or wind turbines) and electric transport expands to include everything from scooters to flying taxis, modern batteries are going to become ever more important. Having the right circuit designs and knowhow will be vital, and I hope you enjoy reading about, or even building, this month's *Battery Manager*.

Matt Pulzer
Publisher

Time for a total rethink?

Techno Talk

Mark Nelson

The way we have electric power delivered to our homes – and the method in which we consume it – needs to be reconsidered. It's inefficient and no longer rational says one think tank. So, what are we doing wrong, and why should we change?

I'm going to hazard a guess. If you are reading this article at home in the UK, then I can say with near certainty that your domestic electricity supply is delivered as alternating current (AC) at not far off 230 volts. Conveniently, your light fittings and electrical appliances are designed for a corresponding 230V AC supply, making for an extremely harmonious setup. But is this ideal? Is it even logical?

Horses for courses

Fifty or even fewer years ago the answer was undoubtedly yes. After all, your appliances (cooker, fridge, washing machine, TV and trendy music centre) and lighting (incandescent and fluorescent) all categorically required high voltages to work efficiently. But a lot has changed in recent decades. We now use LED light bulbs that could operate far more efficiently and effectively from a low-voltage direct current (DC) supply. Ditto for your flat-screen TV and other home entertainment devices. A low-voltage DC supply would be far more appropriate for your computers and other IT stuff, Wi-Fi, security alarm and Internet of Things products. For most people, it will also soon apply to the battery backup supply required to keep your landline telephone and broadband router alive during mains failures. All of these gizmos work inherently on low-voltage DC sources, yet we are still going through the wasteful process of transforming 230V AC down to a likely 12V DC.

Obviously, we still need higher-voltage AC for cooking, heating, refrigeration and clothes washing, but that's as far as it goes. Logic tells us that a dual-supply power system would make better sense and help reduce energy consumption and hence global warming.

No novelty

Thought along these lines is not new, although it has had little exposure or discussion in the media. Nor, as far as I know, has the subject had the possible endorsement of a political party. But that has now changed, with an initiative from the Green Lib Dems, which functions as an environmental think tank for the Liberal Democrat party. You can read full details of the proposal, which was published last November, at: **https://bit.ly/pe-aug22-acdc**

In a nutshell, it argues that according to the Committee on Climate Change, the UK will need a quadrupling of our electrical supply capacity to support electrification of heating and transport. This in turn will require a huge investment in new power distribution infrastructure, which offers a once-in-a-hundred-year opportunity to replace our anachronistic AC power distribution with something much better and truly modern: a DC power distribution system. By establishing the UK as a leader, not a follower, DC switchover can be part of the Green Recovery and a significant export opportunity.

Coexistence is the way forward

The Lib-Dem proposal has plenty of merit, but it is not in harmony with the International Electrotechnical Commission (IEC), which supports maintaining a mixture of AC and DC supply systems, to satisfy the applications that are appropriate for one system or the other. However, the IEC does accept that, 'since the late 20th century, the proliferation of modern power electronics has heralded a growth in decentralised power generation. Coupled with a growing interest in renewable energy sources that generate DC power and the decentralisation of the electricity grid, a new era has begun of co-existence between AC and DC grids.' Emphasising this duality, Jacques Peronnet, Chair of the IEC's Technical Committee 64, declared earlier this year, 'We are moving from a purely AC environment to one that is a combination of AC and DC.'

Do you sodder?

No, it's not a rude word and many *PE* readers indulge in soddering regularly. I confess that I do, although at first, I was unsure whether I would like it or was even capable. It didn't take long to discover that I loved it and I still find the practice positively therapeutic. What's more, I make no secret of this shameless pleasure.

If you are not a sodderer, watch **https://youtu.be/BnbcTtMptkU** where you'll rapidly discover that soddering is how many Americans pronounce (and spell) the word soldering. Alternatively, just consult **www.Urbandictionary.com**, which pompously defines the word as an incorrect term for 'soldering', usually used by idiots and people with 'labtop' computers. I was certainly confused back in the 1970s when hobby electronics shops began selling small plastic containers labelled 'Soder-Wick'. Always curious and easily beguiled by novelties, I asked what Soder-Wick was, and discovered it was a small reel of copper braid that had the amazing ability to absorb and removed unplanned solder splashes on PCBs and other unwanted effects of clumsy soldering. The product wasn't cheap but was much less expensive than a proper solder pump; it saved my bacon many a time.

But why did they call it 'Soder-Wick', and why the missing '*l*'? Was it a misprint or typo? Hardly, because the name was (and still is) a registered trademark. It was many years later that I learnt that 'soder' and 'sodder' are what we Brits call 'solder'. A correspondent on **Quora.com** states that the word *solder* was borrowed from Norman French and, at the time, was spelled without the '*l*'. The '*l*' was added, probably in the 16th century, by people who wanted to make the spelling of French-derived English words match their original Latin spelling. Thus, the traditional pronunciation of the word lacks the '*l*' sound, and this '*l*'-less pronunciation is preserved in American English. (The word derives originally from the Latin verb *solidare*, meaning to fasten together.)

So, our American friends are correct! I feel totally humbled. But I must concede that we Brits are inconsistent over pronunciation. Take the word 'almond'; some people say 'ah-mond', while others say 'al-mond'. Do you call salt 'sollt' or 'sawt'? 'Goll-f' or 'gofe' for the game of golf? It's a crazy, mixed-up world!

Win a Microchip SAM V71 Xplained Ultra Evaluation Kit

Practical Electronics is offering its readers the chance to win a Microchip SAM V71 Xplained Ultra Evaluation Kit (ATSAMV71-XULT) – and even if you don't win, receive a 15%-off voucher, plus free shipping for one of these products.

The SAM V71 Xplained Ultra evaluation kit is a hardware platform to evaluate the ATSAMV71Q21, and other ARM Cortex-M7-based microcontrollers in the SAM V70, SAM S70, and SAM E70 series. Supported by the integrated development platform, the kit provides easy access to the features of the ATSAMV71Q21 and explains how to integrate the device in a custom design.

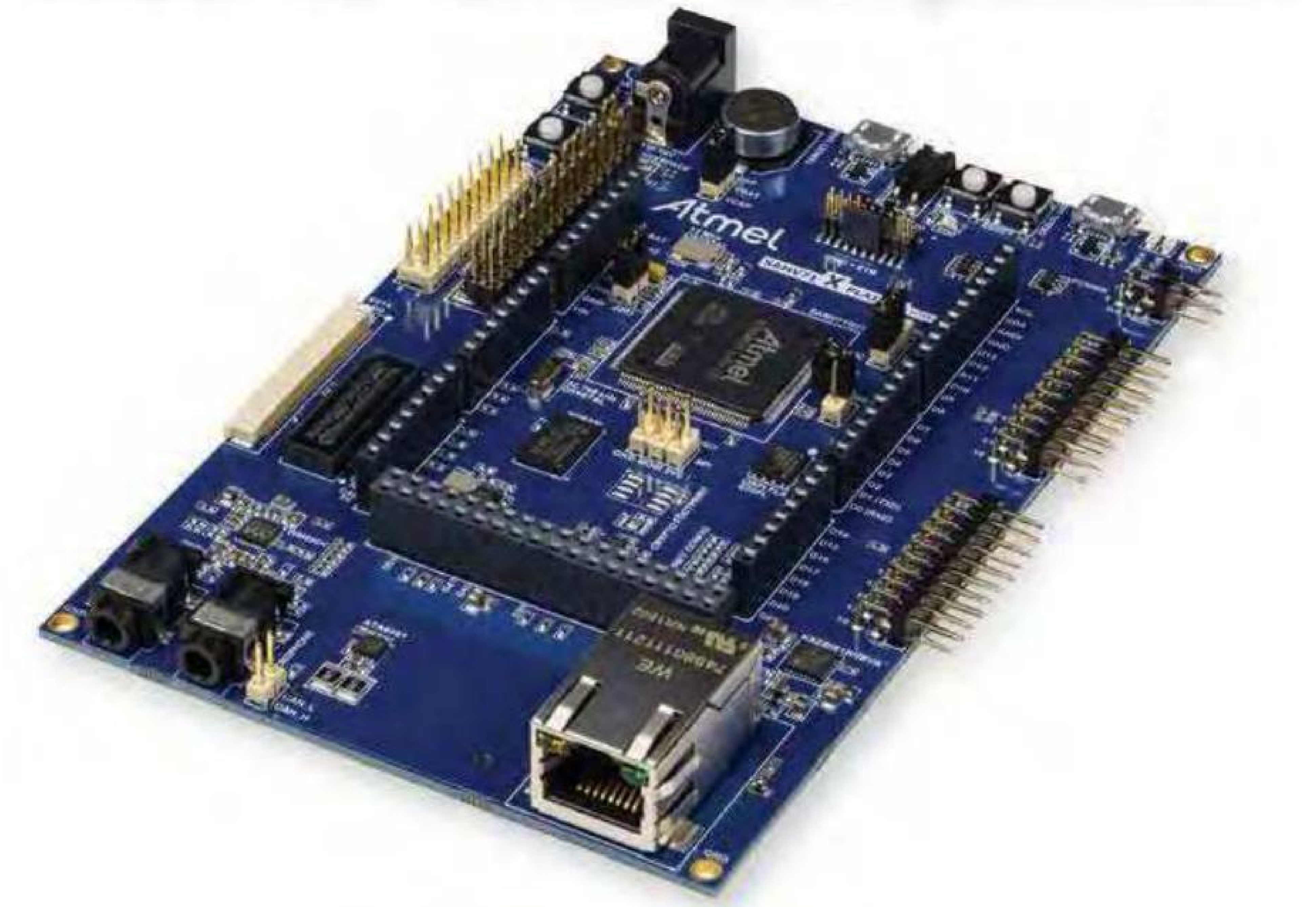
Microchip's SAM V series of Arm Cortex-M7 based automotive microcontrollers focuses on in-vehicle infotainment connectivity for audio amplifiers, telematic control units or head units. SAM V MCUs provide the best combination of connectivity interfaces – including Ethernet AVB, MediaLB, USB and CAN-FD – with the highest-performance Arm Cortex-M core that delivers up to 1500 CoreMarks.

Worth $265.00 (approx £212.50) each

SAM V71 devices are a family of automotive Flash microcontrollers based on the high-performance 32-bit ARM Cortex-M7 processor with a double-precision floating-point unit (FPU). These devices operate at up to 300MHz and feature up to 2048 Kbytes of Flash, and up to 384 Kbytes of multi-port SRAM which uses configurable Instruction and Data Tightly Coupled Memories to leverage the advanced DSP capabilities of the core.

The Xplained Ultra series evaluation kits include an on-board Embedded Debugger, and no external tools are necessary to program or debug the ATSAMV71Q21. The kit also offers additional peripherals to extend the features of the board and ease the development of custom designs.

Free-to-enter competition



Microchip SAM V71 Xplained Ultra Evaluation Kit

How to enter

For your chance to win a Microchip SAM V71 Xplained Ultra Evaluation Kit or receive a 15%-off voucher, including free shipping, enter your details in the online entry form at:

https://page.microchip.com/PE-SAMV71.html

Closing date

The closing date for this offer is 31 July 2022.

April 2022 winner

Malcolm Priestley

He won a Microchip MPLAB ICD 4 In-Circuit Debugger

Net Work

Alan Winstanley

This month we look at how technology is appearing in ever more diverse automotive applications, there's news of updated NFC logos heading our way soon and a brief roundup of EV and space technology news.

A classic 1957 British movie *Hell Drivers* portrayed tipper-lorry drivers duelling on the roads and driving long hours at breakneck speed in order to deliver as many loads as possible and earn a bonus for the day. The 1¾-hour black and white film has some rising British stars and interested readers can currently watch it on YouTube at: **https://youtu.be/7Qb8K1qClyA**

About 30 years after *Hell Drivers* was made, I found myself working for a start-up tech firm that was developing a new electronic in-cab data logger for trucks. This was at a time when HGV (in Britain, 'heavy goods vehicles' – 'semis' or trucks in the US) travel time was recorded physically on wax paper discs in an analogue tachograph, a 'big brother' electromechanical device that many truck drivers initially resented and dubbed 'the spy in the cab'. Truckers disliked the idea of being watched.

Legislation stated that drivers could drive for nine hours before parking up for the night, but they also had to take a break after 4½ hours at the wheel. Once back at the depot, paper discs were checked for legal compliance, a chore that was often sub-contracted out to third parties. Our new in-cab computer promised to eliminate this paper-based technology; it connected to a nondescript port (which we reverse-engineered) on the back of analogue tachographs, and it captured the journey data and wrote it onto our removable memory key. The key would plug into the office PC, where our software instantly analysed it. It was cutting-edge, trend-setting stuff, but sadly it was probably too far ahead of its time.

Today, digital and 'smart' tachographs are available to monitor journeys, and data is stored both in the unit's internal memory and also on compulsory driver smart cards. To show what onerous regulations today's truck drivers have to contend with, these are summarised on the UK Government website (see: **https://bit.ly/pe-aug22-tacho**). Both analogue (paper disc) and digital systems are covered, and you can also see what traditional paper discs look like.

Black box data

One forward-looking feature of our 1980s truck computer was the idea of capturing data in case there was an event such as (heaven forfend) a road accident. If sudden deceleration was detected, the last few minutes' worth of data could be telescoped-out for detailed analysis afterwards. This principle is used today in certain dashcams, where G-force sensors cause data to be written to a read-only folder in memory, ready for later examination.

The same kind of technology also benefits today's younger motorists, or those with poor driving records. Telematics ('black box') units fitted inside cars, or smartphone apps, monitor the driver's general style and habits, and coupled with GPS location tracking, they are designed to encourage safe and steady driving and keep insurance costs down.

Telematics data is fed back to motorists regularly: too much heavy-footedness or speeding can see premiums rise or insurance policies being withdrawn altogether. This way, drivers become conditioned into bettering their road skills. Some telematics policies impose a curfew that prohibits night-time driving, when accidents are more likely to occur. Younger drivers may even be forced to share data with their parents! Many car insurers offer telematics policies and there is plenty of material on YouTube explaining how they work and what it's like to live with one. They may also assist older, steadier drivers to cut their insurance premiums.

Safety drive

Many modern cars are peppered with electronic driving aids that at one time would have appeared only on expensive, high-spec vehicles. Drivers today can enjoy rain-sensing windscreen wipers that operate when an infrared

VDO
R 01.04.2015 13:00
01.04.2015 15:45

A modern digital tachograph with smartphone management (Courtesy Continental AG).

A telematics device fitted to a car windscreen (Image: Adrian Flux Insurance / YouTube).

A swarm of Sky Magic drones being prepared for take-off.

Sky Magic drones created an image of a corgi dog – the Queen's favourite pet (BBC TV).

light path is disrupted by rain or snow; headlight main beams that dip before a driver can react; warnings of vehicles entering the driver's blind spot; 'cross traffic alerts'– great when reversing out of car park spaces and a huge van has blocked the driver's view. Then there's traffic sign recognition, which the author is finding increasingly useful in today's busy roads, and 'Lane Keep Assist' which detects when lane markings are crossed (a nuisance that I soon switched off). It all points to increasingly sophisticated sensors and AI, 'reading' the road ahead.

Some aids are being adopted by law in Europe, starting with Intelligent Speed Assist (ISA), the traffic sign and/or GPS speed limiter that I've mentioned in earlier columns. Advanced Emergency Braking (AEB) will both sense the proximity of traffic ahead and brake the car automatically. Being launched onto an unsuspecting public is the controversial rule that new cars (2022 on) must also be 'compatible' to allow a breathalyser to be fitted. I suspect that new cars will just carry some extra wiring, rather than being fitted with the breathalyser itself.

The journal *Euro Weekly News* points out another safety aid aimed at new European cars this year: the 'fatigue and sleep detection assistant' which gauges the driver's steering movements, looking for constant or erratic corrections while measuring the length of time the engine's been running. It's speculated that sensors could even test the pressure drivers exert on the steering wheel; indeed, trials involving electrocardiograms embedded in steering wheels were held nearly a decade ago (see: **https://bit.ly/pe-sug22-ecd**). The purpose of this elaborate technology is merely to light up a dashboard 'coffee cup' signal reminding weary drivers to take a break. I promise I am not making this up – how VW sees the Driver Alert System working is shown in a YouTube video at: **https://youtu.be/WqR3YbjxyS0**

Magic in the Sky

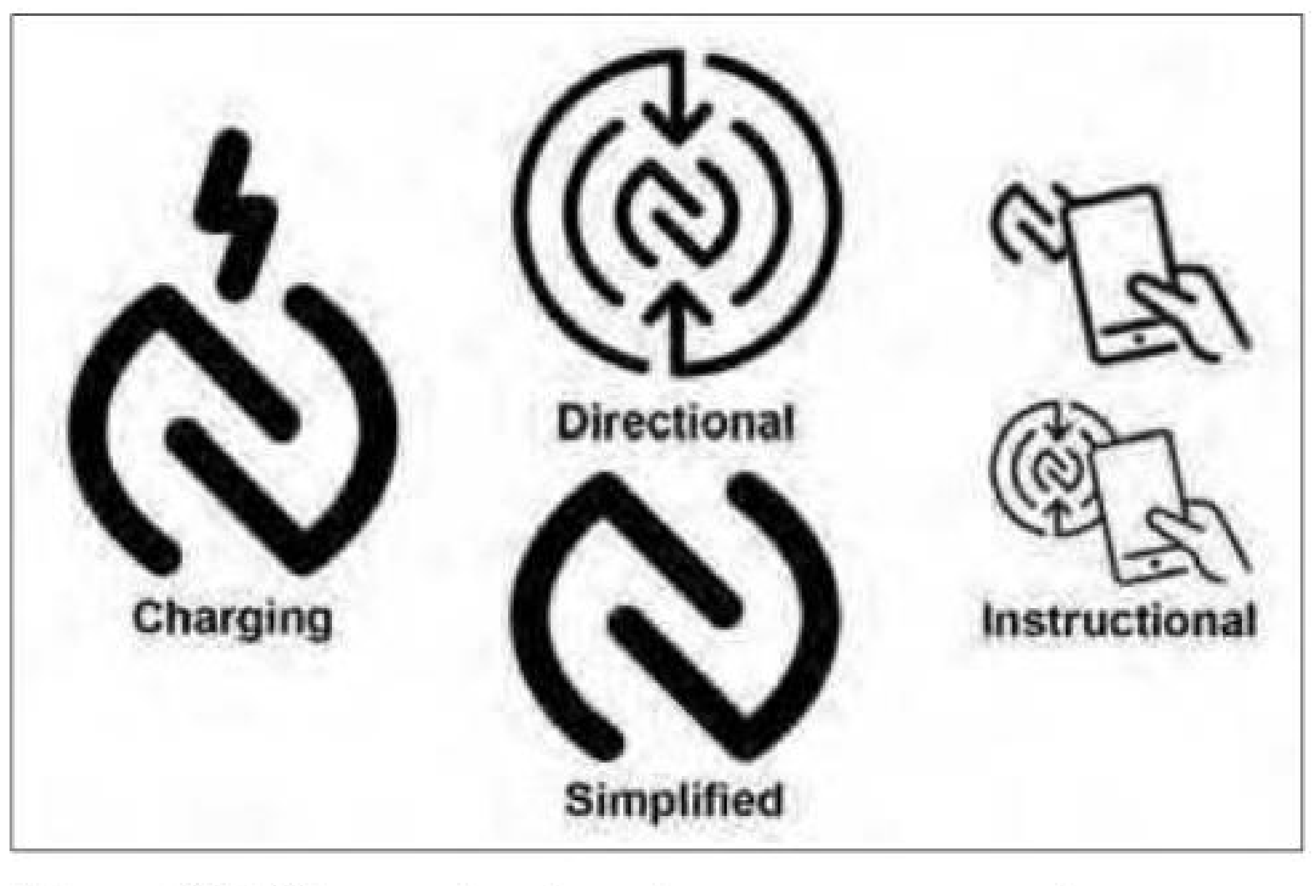
Our overseas readers will hopefully forgive us if, as a British magazine, we take time out to celebrate the marvellous once-in-a-lifetime spectacle that was the Queen's 70th Platinum Jubilee held in London a few weeks ago. It brought out the best of British pomp, ceremony and humour, and the various processions were organised with astonishing efficiency as everyone joined together to celebrate 70 years of Queen Elizabeth II's rule. Apart from the carnival's float parades, another marvellous feature was the night-time light show depicting various images in coloured lights, carefully co-ordinated in the skies over London. We were treated to images of the Queen's favourite corgi dog, prancing horses, a pot of tea being poured, a graceful swan and of course the Queen's handbag, all skilfully choreographed in a faultless display over Buckingham Palace.

The creative genius behind the light show is London and Singapore-based Sky Magic, which specialises in creating drone displays for indoor and outdoor events. Their work has appeared in Sydney, Dubai, Venice and many more locations besides. Sky Magic's custom drones incorporate nine super-bright LEDs producing a total of 900 Lumens that are visible from 2km away. They're optimised to reduce power consumption and maximise flight duration in displays that span some 300m in width. A swarm of 400 computer-controlled light-emitting drones was marshalled for the Jubilee display. A strong emphasis is placed on flight safety/security, they say, and their control software generates smooth collision-free trajectories for each drone. The aerial festival was an enthralling experience, and interested readers can see more about the technology at: **https://skymagic.show**

The *Party at the Palace* TV programme is available on BBC iPlayer for UK-based, or (overseas) VPN users accessing a UK-based server, at: **https://bbc.in/3n1l3D9**

Whose IDea is this

Last month's *Net Work* included an item on NFC (Near-Field Communication), the contactless 'tap' technology found in smartphones that is used for cashless payments and a whole lot more. NFC radio tags are more convenient to use than a QR code, as they don't need an app or camera to scan an image, but sometimes problems arise if NFC users aren't sure *where* to tap their phone. To overcome this, a new set of symbols will gradually appear on NFC-enabled equipment. Industry is being encouraged to select the most appropriate NFC 'Wayfinding' symbol, depending on a user's



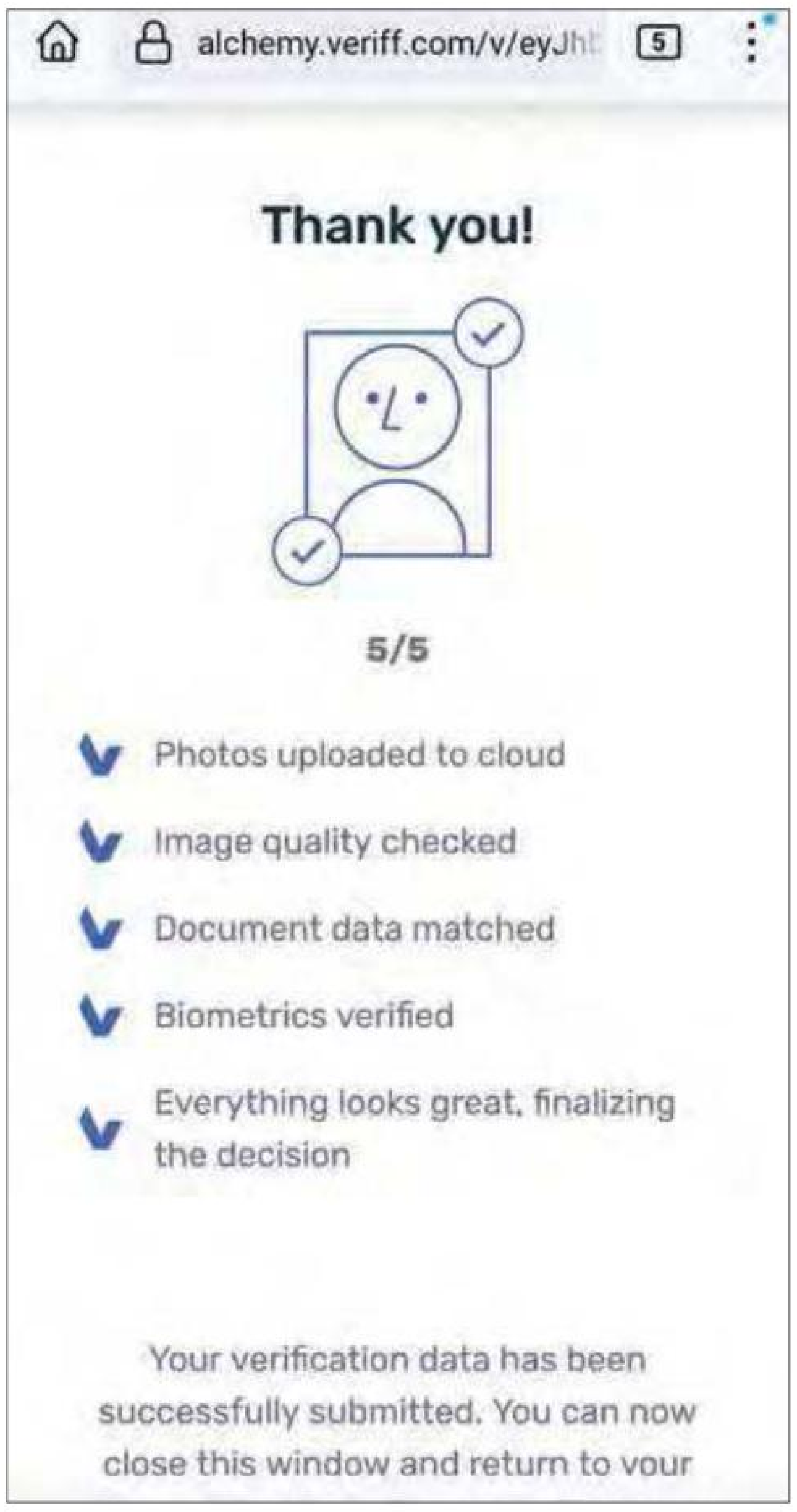
New NFC symbols show users where and how to tap their NFC-enabled device (NFC Forum).

HYPERLOOPTT

A Hyperloop 'capsule' under development in Toulouse, France.

likely familiarity and experience of the equipment and how often a tag would ever be tapped (once or twice in a lifetime? All the time?). There's more at: **https://youtu.be/QZEftIt2fbE**

One buzzword bandied around in commerce is 'KYC' or *Know Your Customer*. How do online traders or other bodies know it's really you tapping away on the keyboard? How do they guard against, say, money laundering or online fraud? Recently, I was asked to verify my own ID by an online merchant using the Veriff ID checking service. A web page QR code displayed a URL to open with my smartphone, after which the Veriff applet popped up. I had to scan both sides of my UK driving licence before submitting a selfie. (A passport with built-in NFC chip could also be used: the smartphone will scan the NFC chip directly.) My details were uploaded to a cloud somewhere, and a minute later I was duly 'verified'.



Thank you!

5/5

- Photos uploaded to cloud
- Image quality checked
- Document data matched
- Biometrics verified
- Everything looks great, finalizing the decision

Your verification data has been successfully submitted. You can now close this window and return to your

Verifying one's ID using the Veriff system requires a scan of photo ID and a selfie.

To be fair, it couldn't be easier, and the system worked flawlessly, but the process highlighted how our personal data gets bandied around, and one might wonder what happens to the flow of our private data behind the scenes or where it ends up. There's no suggestion that it's insecure, but from the citizen's perspective, I found the process was anything but transparent.

I also wrote last month about the pitfalls of society sleepwalking into becoming a cashless culture, one that is beyond the reach of those who don't have, or can't use, a smartphone or a multitude of apps. Someone who is infirm, elderly or doesn't own a smartphone will struggle to access online services in the first place, but as Baroness Altmann wrote recently in the *Daily Mail*, 'From car parking to banking and GP visits, we must stop punishing the elderly for the crime of not being able to work an app.' She added that 'three million people in Britain did not access the internet — more than two million of them aged 70-plus. Even ten percent of mobile phone users don't own a smartphone, denying them the capacity to download the 'apps' with which so many companies seem to be entirely obsessed and increasingly foist on their customers, in an effort to cut costs and increase their profits. ... As a result, a large section of society is left feeling at best marginalised, at worst completely forgotten.'

Quite right. Even those who *can* use smartphones and apps today will eventually find themselves neither interested enough nor able to do so, but the slow drift towards relying on mobile apps and tiny touchscreens to conduct everyday business seems to continue unabated.

Prepare for take-off

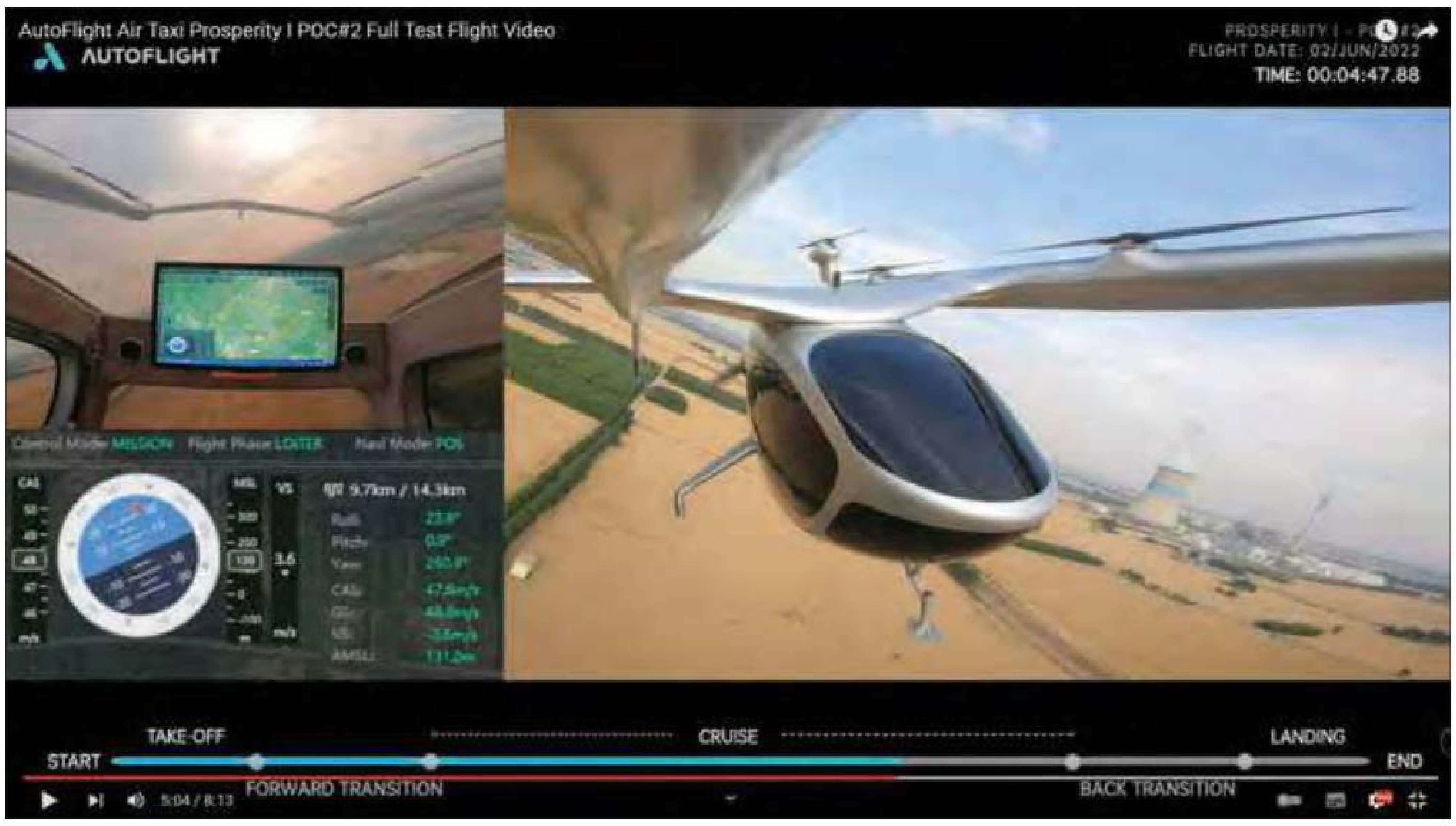
Decades ago, old retail stores and offices used pneumatic tubes carrying little capsules to shift cash or paperwork rapidly and securely between locations. 'Money air tube systems' are still produced by Aerocom, for example (**www.aerocom.co.uk**), evidence that we haven't quite gone cashless yet! Fast forward to today, and research and feasibility studies continue on a proposed 'hyperloop' transport system, which aims to move passengers and freight safely over ground in capsules travelling at up to 750mph – faster than a Boeing 777. No pneumatics here – linear motors will drive the pressurised carriages on a magnetic cushion, contained within low-pressure tubes.

Hyperloop Transportation Technologies (HyperloopTT, see: **www.hyperlooptt.com**) is headquartered in the US and Toulouse, France and has now received investment from the scientific and engineering consultancy Thornton Tomasetti to take the project forward. While Britain stumbles to ever complete its HS2 high-speed rail line of just a few hundred miles (see: **www.hs2.org.uk**), a hyperloop overground pipeline-based transport network could see futuristic seaport and 'hyperport' terminals being built in the US or across Europe. I'll track the progress of Hyperloop, and an interesting insight is at: **https://youtu.be/uwm3qvFWVRU**

AutoFlight's electric vertical take-off / landing (eVTOL) *Prosperity* air taxi featured earlier in *Net Work* columns successfully completed its second proof-of-concept test in June. A full-size prototype with revised propeller configuration has made 30 flights, transitioning from vertical lift to forward cruising and back home again. The air taxi will carry three passengers plus a pilot, and a production-ready version of the air taxi is expected later this year. Drone enthusiasts will be interested in a full-test-flight video at: **https://youtu.be/o3t40KcnMBo**

According to their latest *Mobility Consumer Index*, 52% of motorists planning to buy a car will now choose an EV, said analysts Ernst & Young in a March 2022 survey, in what was described as the 'tipping point' for the 'electric vehicle' market. The internal combustion engine is not dead and buried yet, though; digging down, I found that 'electric vehicles' included hybrid (HEV) and plug-in hybrid (PHEV) cars which of course contain petrol or diesel engines too. Buyers cited 'environmental reasons' as the main factor when choosing an EV, the survey said, yet only 20% of them would choose a battery electric vehicle (BEV); the rest are happy with hybrid fossil-fuelled cars.

Latest figures from the UK's Society of Motor Manufacturers and Traders



AutoFlight's Prosperity eVTOL air passenger taxi completed a second round of trials in Germany in June.

(SMMT) show electric vehicle sales rising, with 20% of new registrations being EVs. This still represents just 2% of all vehicles on British roads. At a time when petrol (gasoline) and diesel fuel costs are at an all-time high, 20.5 million petrol cars and 13 million diesels make up 95% of Britain's car ownership, says the SMMT, admitting there is 'significant ground to cover if we are to fully decarbonise road transport "at pace". This is coupled with the first consecutive annual fall in vehicle numbers in more than a century... leading Britons to hold onto their cars for longer'. Yet our TV screens are stuffed full of adverts for space-age electric cars at prices that are out of this world.

Previously mentioned in *Net Work* (May 2022) the start-up EV maker Fisker tells me it now has 50,000 reservations for its Fisker Ocean SUV. The company has released details of its second model, the Fisker PEAR, described as an 'agile urban EV'. The new Fisker PEAR has over 3,200 pre-orders, with production expected to begin in 2024. More details at **fiskerinc.com**

Meantime, high-end EV brand Polestar has partnered with StoreDot, mentioned in earlier columns, with an investment aimed at exploring StoreDot's extreme fast charging (EFC) battery technology for use in future models.

Other news

Windows PC users are warned of a surge in deadly ransomware attacks that exploit a Zero-day security hole – the so-called 'Follina' vulnerability – in Microsoft's operating system. This latest wave of malware can wreck networks, so users should be extra-vigilant and watch out for dodgy-looking MS Office or rich text files, .zip files arriving by email or .html links (eg in forums) that may download malware onto your system. Particularly worrying is that simply 'Previewing' infected files in File Explorer's Preview pane can trigger the attack, even without opening the file, so Preview should be disabled, at least until the flaw is patched in a Windows update.

How ransomware gangs are clubbing together to attack networks is published on *The Register* IT website at **https://bit.ly/pe-aug22-zero** which also explains how machines can be infected by the Follina exploit in the first place. A detailed analysis has been published by Sophos at: **https://bit.ly/pe-aug22-ms**

In June, Blue Origin, the sub-orbital space rocket firm owned by Amazon's Jeff Bezos, completed its fifth human spaceflight and the 21st flight for the New Shepard program (see *Net Work*, March 2020.) Blue Origin's New Shepard booster rocket lands vertically, leaving the crew capsule to parachute down to Earth separately. More at: **www.blueorigin.com/new-shepard/fly/**

Instrumentation on NASA's James Webb space telescope continues to be commissioned ready to commence operation. One of the observatory's mirrors was recently struck by a widely reported, but tiny micrometeoroid which has not affected the telescope's operation. Its first full-colour images are due in mid-July and the web page **https://go.nasa.gov/39CT6Pd** shows the current progress, along with the instrument's operating temperatures – it's rather chilly at –241°C.

Sony has established a new division called Sony Space Communications Corporation to develop high-speed laser beam communications devices for microsatellites flying in low-earth orbit. The miniaturised technology is aimed at microsatellites where space is at a premium (no pun intended), and will enable real-time communications from anywhere on the ground up to any satellite in space, as well as between satellites themselves. The use of optical techniques also avoids the complexities of radio frequency licensing, Sony adds.

That's all from *Net Work* this time, see you next month!

The author can be reached at:
alan@epemag.net

HAMMOND

new!

1551W IP68 miniature enclosures

Learn more: hammfg.com/1551w

Contact us to request a free evaluation sample.
uksales@hammfg.com 01256 812812

Multi-purpose Battery Manager

By Tim Blythman

Our recent *Battery Multi Logger* is a great tool for monitoring and diagnosing battery problems. But sometimes, you need something which will not just monitor what's going on but also take action, such as connecting and disconnecting loads based on battery charge state. That's just part of what this *Battery Manager* does.

Our Battery Multi Logger (*PE*, February-March 2022) is a Micromite-based device that monitors the condition and usage of a battery system. It can handle battery systems between 6V and 100V, and it is a convenient tool to keep track of how batteries are being used, ensuring that they are kept healthy.

Being heavily discharged or overcharged can greatly reduce a battery's working life, possibly leading to the need to buy an expensive replacement prematurely. So you want to be sure that you're treating them well.

Both of these conditions are relatively easy to rectify, as long as you are aware of them happening, by simply disconnecting the load(s) or charger causing the problem.

Our recent *Battery Multi Logger* unit can monitor this but did not have any means to take corrective action – until now. The *Battery Manager* adds switching modules to the *Battery Multi Logger*, which can connect or disconnect loads and chargers to keep the batteries healthy.

Part of the design is a new I/O Expander board that provides control signals to allow up to four Switch Modules to independently and automatically connect and disconnect loads as needed.

The *Battery Manager* can also interface with the *High Current Four Battery Balancer* (*PE*, April-May 2022) to provide even more detailed information about the state of a multi-cell battery or multi-battery system. The *Battery Manager* can even be used to program, control and monitor the *Battery Balancer*.

While the *Battery Multi Logger* hardware remains mostly unchanged from the published design, a new control program adds the interface to configure, control and monitor the Switch Modules and *Battery Balancer*.

The Switch Module and I/O Expander hardware also have uses outside the *Battery Manager*. While designed for 3.3V operation, the I/O Expander board will happily work at 5V, so it could be hooked up to an Arduino board or just about any other microcontroller.

Features

- *Compact, flexible and modular addition to the Battery Multi Logger*
- *Connect and disconnect up to four loads/sources to protect batteries*
- *Low battery drain*
- *Can interface with the High Current Four Battery Balancer*
- *Capable of switching well over 20A (possibly over 30A) at 10V-60V*

Similarly, the Switch Module will work with just about anything that can supply a control signal of 3.3-15V. So it can also be driven directly by just about any microcontroller.

Switch Module

One of the goals of the *Battery Multi Logger* is to use as little power as possible. So we have designed the new Switch Module to have very low quiescent and operating currents.

We are using high-current MOSFETs as switches because these can be controlled with minimal power. The MOSFETs are driven by a latching circuit that 'remembers' the state of the switch and drives the MOSFET gates on or off as needed.

This latch can be toggled in several different ways. A pair of switch contacts connected to the latch circuit can set its state, providing simple pushbutton control.

The Switch Module PCB also incorporates a pair of opto-isolators. Their output transistors are in parallel with the switch contacts. Thus, there is also the option to set the latch state and control the MOSFETs via an electrically isolated interface.

I/O Expander board

You might recall from the *Battery Multi Logger* article that it doesn't have many free I/O pins left. The two pins that provide the COM1: serial port are not used, though, and are brought out to the *Battery Multi Logger* PCB edge.

But we have earmarked these to interface with other serial devices. A better way to control Switch Module(s) is to use the I^2C interface, which is brought out to pins at CON4 of the *Battery Multi Logger* PCB.

We are using a PCF8574 IC, which we described in our article on I/O Expander Modules (*PE*, November 2021). This lets us easily add eight I/O ports. In fact, with multiple ICs, we could add up to 128 I/O ports, although that would exceed our requirements.

So we have designed a small I/O Expander PCB, which can be controlled using the available I^2C bus. It provides eight I/O pins connected to transistors to drive the opto-isolated inputs of th Switch Module(s).

As noted above, a low quiescent current is important. The PCF8574 draws around 10µA when there is no activity on the I^2C bus. Its primary current consumption is the current it supplies to drive the opto-isolators, and they are only active very briefly during switching.

Battery Balancer support

As we just noted, the COM1: serial port on the *Battery Multi Logger* is free for us to use. Since the *Battery Balancer* already has a serial interface, we can simply connect these to allow communication and control between the two.

We can also use the *Mini Isolated Serial Link* (*PE*, March 2022) to isolate the different parts of the system.

Fig.1 shows an overview of the additions to the *Battery Multi Logger* to turn it into a *Battery Manager*. Note the connection from the *Battery Balancer* to CON6 on the *Battery Multi Logger*.

Updated software

Naturally, these extra features need to be controlled and configured. This is done via extra buttons and pages on the *Battery Multi Logger's* Micromite LCD interface, shown in screengrabs later in this article. There is also a more detailed description indicating how to use these new screens along with those images.

The first new page controls the Switch Modules; up to eight triggers can be set. These are voltage or current thresholds that result in an action occurring, such as one or more of the Switch Modules being activated. A latch is also set to prevent repeated activation; a trigger can also reset a latch to provide alternate operation.

For example, Trigger 1 can be set to activate when the battery voltage falls below 11V. This sets Trigger 1's latch and, via a Switch Module, also disconnects some non-essential load from the battery, reducing the chance of damage to the battery from deep discharge.

Trigger 2 is set to activate when the battery rises to 12.5V and also to reset Trigger 1. Similarly, Trigger 1 can reset Trigger 2. As you might expect, Trigger 2 would be configured to reconnect the load that is disconnected by Trigger 1.

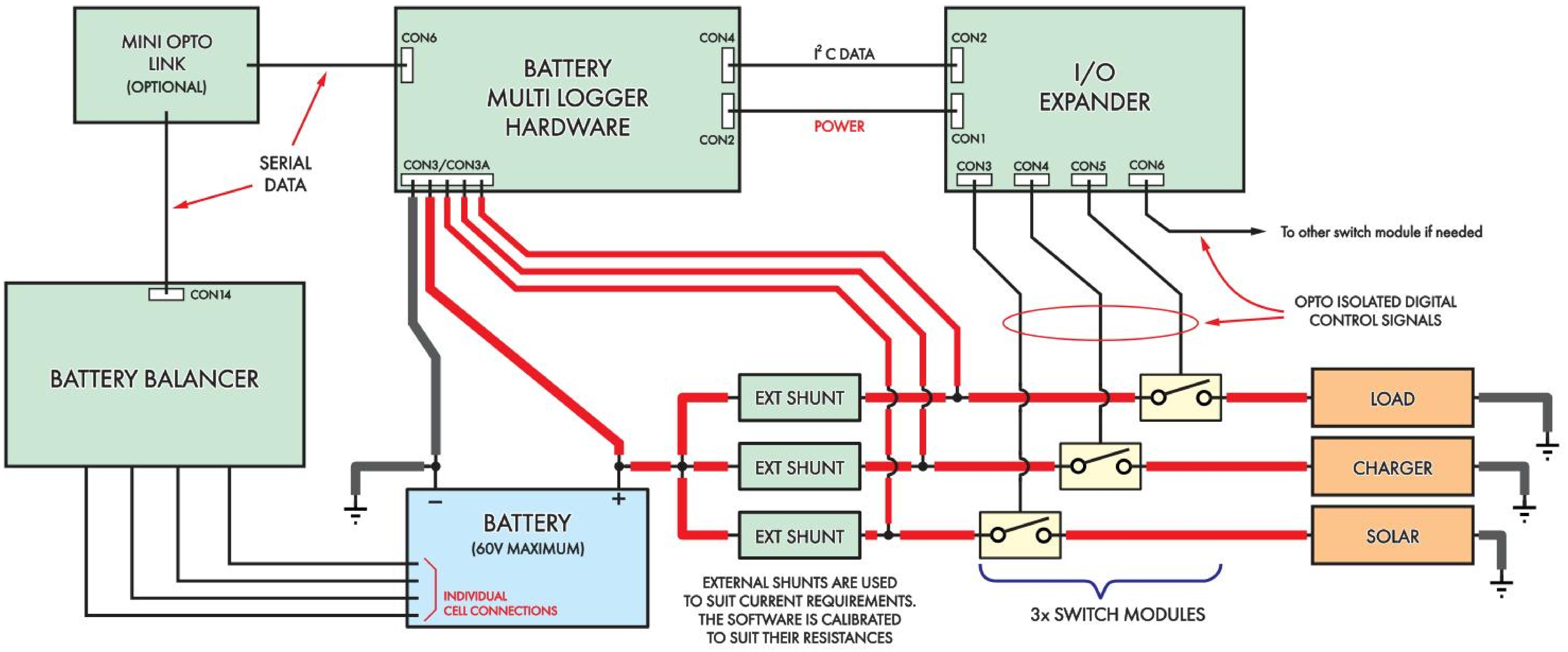

MINI OPTO LINK (OPTIONAL)
SERIAL DATA
CON6
BATTERY MULTI LOGGER HARDWARE
CON4
CON2
CON3/CON3A
I²C DATA
POWER
CON2
CON1
I/O EXPANDER
CON3
CON4
CON5
CON6
To other switch module if needed
OPTO ISOLATED DIGITAL CONTROL SIGNALS
CON14
BATTERY BALANCER
EXT SHUNT
EXT SHUNT
EXT SHUNT
LOAD
CHARGER
SOLAR
–
+
BATTERY (60V MAXIMUM)
INDIVIDUAL CELL CONNECTIONS
EXTERNAL SHUNTS ARE USED TO SUIT CURRENT REQUIREMENTS. THE SOFTWARE IS CALIBRATED TO SUIT THEIR RESISTANCES
3x SWITCH MODULES

Fig.1: the Battery Manager consists of the Battery Multi Logger plus the peripherals shown here. Up to four Switch Modules can be added with one I/O Expander – we imagine most constructors will need one or two. It can also interface with the High Current Battery Balancer, allowing cell status and balancing activity to be monitored.

Thus these two triggers work to detach a load from the battery except when it has sufficient charge. A similar arrangement in reverse can also work as a crude charge regulator, preventing overcharging.

The external switches can also be manually manipulated, either for testing or to override the programming; plus you can also manually reset the triggers.

Another page shows the current operating state of the *Battery Balancer* (as reported by the *Balancer* over its serial port), including which cells are being balanced, in which direction and to what extent. Buttons are also provided to issue commands to the *Battery Balancer*.

Two graph pages are available to show recent data from the *Battery Balancer*. One page shows the cell and stack balancing activity, while a second page plots the individual cell voltages.

I/O Expander operation

The circuit diagram of the I/O Expander module is shown in Fig.2. Its CON1 header connects to the *Battery Multi Logger's* CON2 for 3.3V power and ground. The I^{2}C bus is present at the *Battery Multi Logger's* CON4, which connects to CON2 here.

These are situated to align directly, allowing the I/O Expander module to stack onto the existing hardware, using either stackable headers or being directly soldered. Since it is only four wires, it can be run remotely too, although a direct connection is preferable.

The *Battery Multi Logger* hosts the pull-up resistors required for the I^{2}C bus, so these are not present on the I/O Expander board. It's generally better for pull-ups to be on the master, since only one pair is needed per bus.

On the I/O Expander PCB, the 3.3V, ground, SDA and SCL lines from CON1 and CON2 go to IC1's pins 16, 8, 15 and 14, respectively. A 100nF capacitor bypasses IC1's supply. There are two more 100nF capacitors to help source current into downstream connectors CON3-CON6.

IC1, the PCF8574, has three address pins (1, 2 and 3) that need to be pulled up or down to set its address. We avoid the use of pull-up or pull-down resistors as this could increase current consumption. So a group of three jumper pads, JP1-JP3, is provided for this purpose. All pins are pulled low by default, giving a 7-bit address of 0x20 hexadecimal (32 decimal).

These jumpers are actually solder pads on the PCB and can be changed by cutting the thin traces and soldering between pads. Since eight I/O pins are ample, we have written the software to simply work with a single I/O Expander board with the default address.

The PCF8574 could be replaced by the mostly identical PCF8574A. The only difference is that the PCF8574A uses a different range of addresses; in this case, the default address will be 0x38 (56 decimal), and the software would need to be modified to suit that value.

Pin 13 of IC1 provides an active-low input change interrupt signal, which is not used in this application. We are using all of the pins as outputs, so we do not need the interrupt function.

The remaining pins labelled P0-P7 (pins 4, 5, 6, 7, 9, 10, 11 and 12, respectively) are the I/O pins. They are either weakly pulled up (the default state) or pulled low by a sink capable of around 10mA.

Since the opto-isolators on the Soft Switch are active-high devices, we use P-channel MOSFETs controlled from these I/O pins to source current from the 3.3V rail. These MOSFETs also invert the signals.

For example, Q1's gate is connected to pin 4 of IC1 (P0). The gate is also pulled high by a 10kΩ resistor. While probably not strictly necessary, we have fitted these so that false triggering does not occur while the *Battery Multi Logger* is powering up.

Q1's source is connected to the 3.3V rail and is effectively connected to its

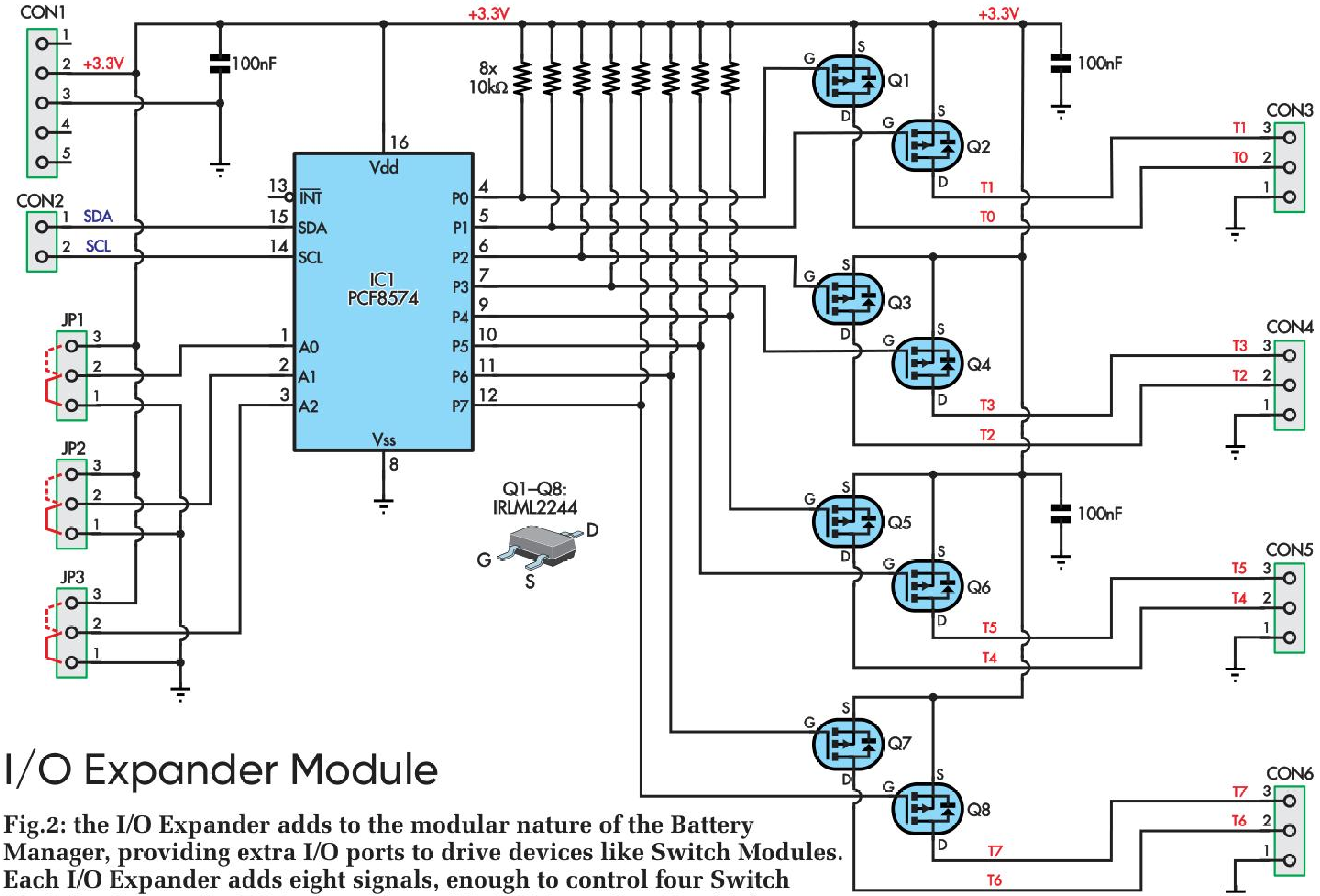


Fig.2: the I/O Expander adds to the modular nature of the Battery Manager, providing extra I/O ports to drive devices like Switch Modules. Each I/O Expander adds eight signals, enough to control four Switch Modules. It uses the PCF8574 addressable I/O Expander IC, which can be configured to respond to eight different addresses, allowing further expansion.

drain when the gate goes low, delivering 3.3V to pin 2 of CON3.

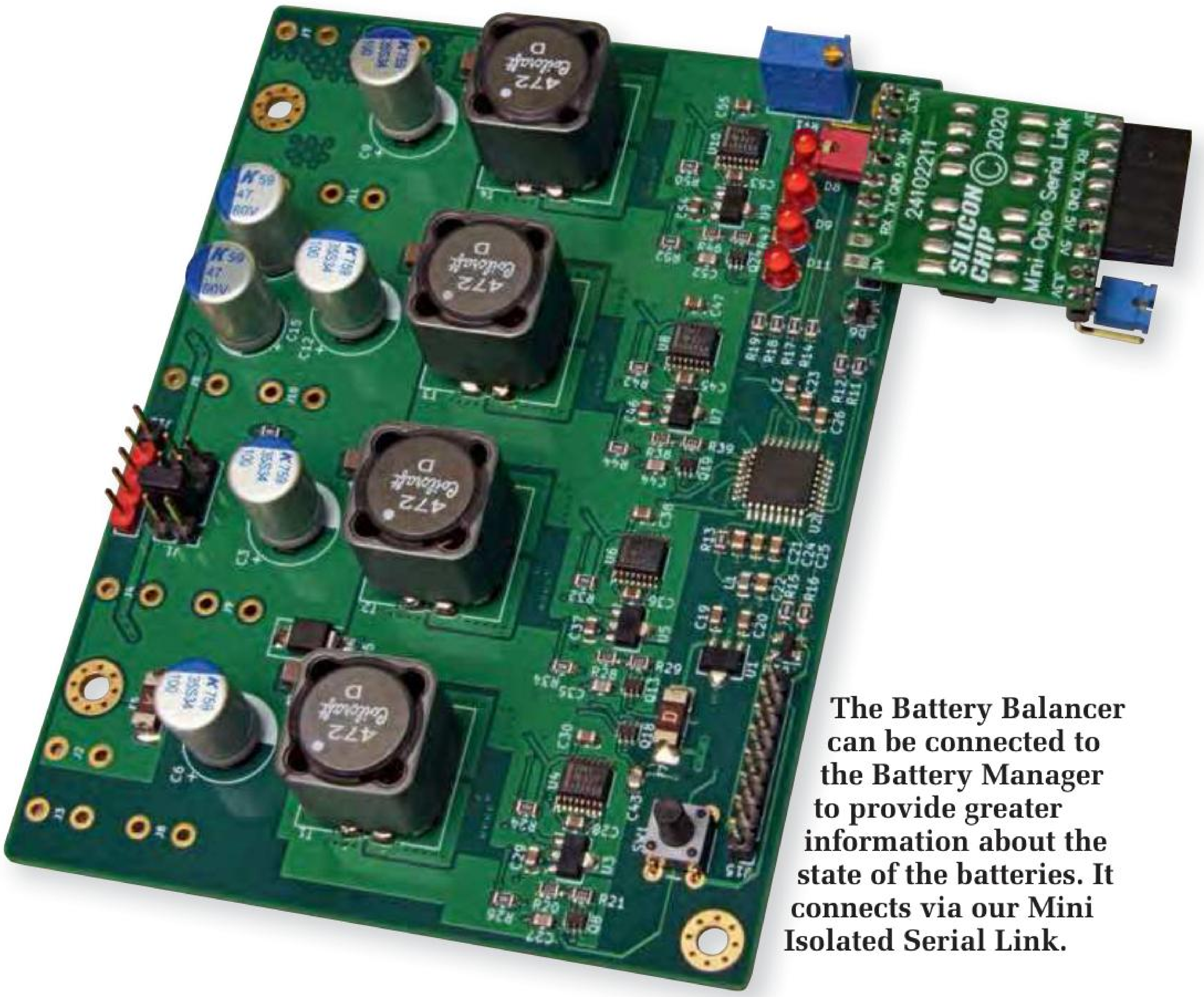
A similar arrangement exists for the other seven I/O pins of IC1. The outputs are arranged in pairs, to provide the complementary on/off functions needed for the Switch Module to operate.

Each of CON3-CON6 can connect to the input of a Switch Module, and so we can control up to four Switch Modules with one I/O Expander board.

In operation, the I^{2}C host (in this case, the *Battery Multi Logger*) writes a default value of 0xFF (all bits set high) to IC1, which then sits in this idle state, drawing virtually no current. Its output transistors are off, and all pins on CON3-CON6 are not connected to the 3.3V rail.

When an output needs to be activated, the *Battery Multi Logger* sends a data byte with at least one bit set low. This causes the corresponding pin from P0-P7 to go low, turning on its MOSFET and sending its corresponding output high.

For the brief period that the I^{2}C bus is active, IC1 draws a modest 100µA, while any of P0-P7 that are active will cause less than 1mA to be sunk through its pull-up resistor. The transistor will also source whatever current is needed to control the connected Switch Module.



The Battery Balancer can be connected to the Battery Manager to provide greater information about the state of the batteries. It connects via our Mini Isolated Serial Link.

Switch Module operation

Fig.3 shows the circuit diagram of the Switch Module. As mentioned, up to four Switch Modules can be connected to a single I/O Expander board.

CON1 and CON2 are large, high-current connections to the positive end of a battery and its load or source (eg, a charger). Thus, we perform high-side switching, leaving the ground connections uninterrupted. The connections are not polarised, so current can flow in either direction when the switch is on.

Across CON1 and CON2 are connected pairs of back-to-back P-channel MOSFETs, Q4-Q11. Their sources are connected together, with the drains going to either CON1 or CON2. With the gates held near the source potential, the transistors do not conduct, and the switch is off. If the gates are taken low relative to their sources, then a low-resistance path exists between CON1 and CON2.

The MOSFET body diodes pass a positive voltage from either CON1 or CON2 to the remainder of the circuit. CON3 is used to provide a ground connection for the circuit and is wired to the battery system's common negative terminal.

The 10kΩ resistor in series with the GND connection and the 100nF capacitor across ZD2 provide a filtered logic supply (between LOGIC+ and LOGIC_GND). Typically, around 90% of the battery voltage is present across the 100nF capacitor and ZD2.

Zener diode ZD2 does not conduct under normal conditions; it is not even strictly needed for 12V systems, but will clamp any spikes that might be present. It also allows the switch module to be used with battery voltages over 60V.

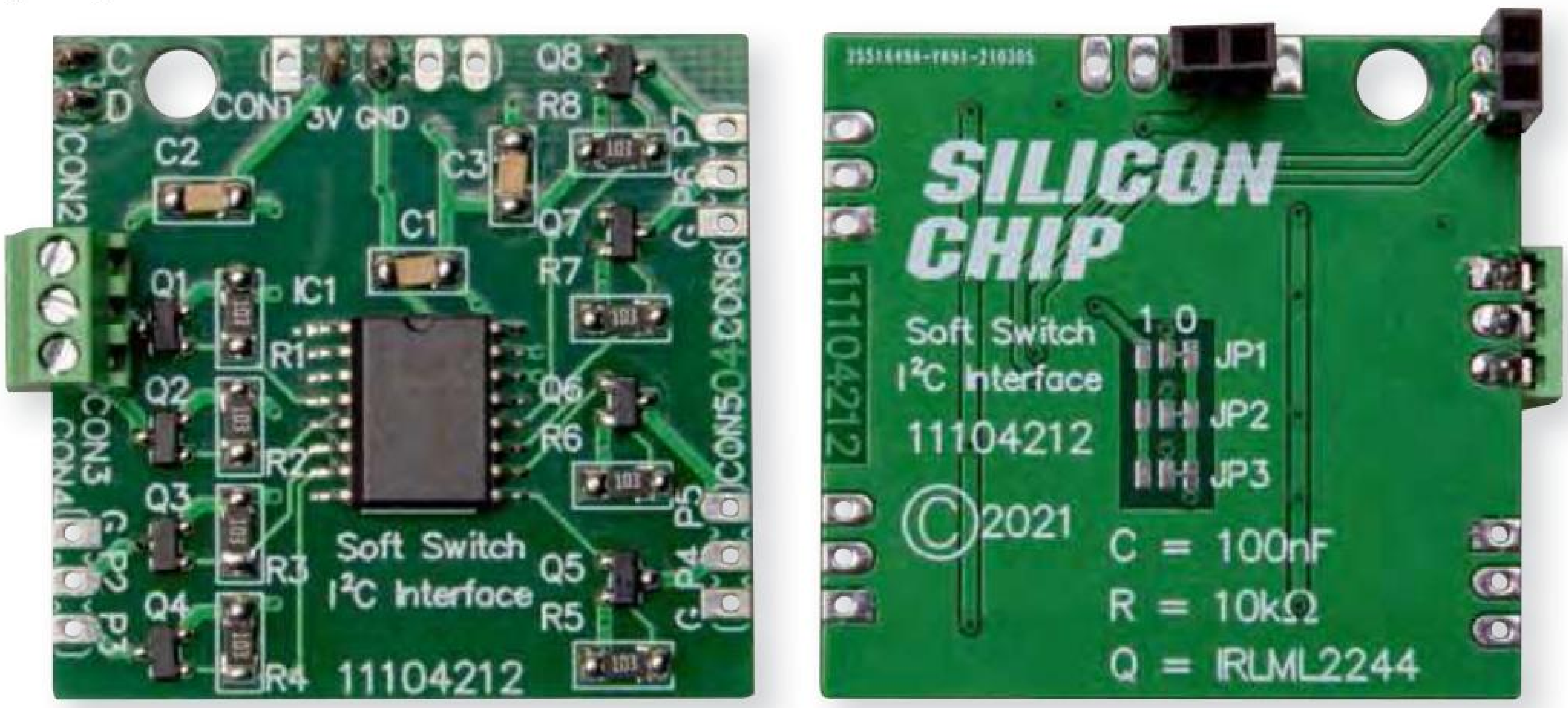
Q1 and Q2 are configured as a bistable latch, with the 100kΩ and 220kΩ resistors connected to their gates – these provide a mutually exclusive interlock.

The gate of Q1 is connected to the drain of Q2 and vice versa. If Q1 is on, then Q2's gate is pulled to near its source voltage, and it is off. Similarly, if Q2 is on, then Q1 must be off. This latch is what retains the state of the Switch Module.

Q1's gate is also connected to Q3's gate, so that Q3's state is generally the same as Q1's. Q3's drain is also connected to Q4-Q11's gates. When Q3 is on, its drain network (consisting of the 100kΩ and 220kΩ resistors and 15V zener diode ZD1) drives the gates of Q4-Q11 to 4-15V below their sources.

In this state, Q4-Q11 turn on, closing the Switch Module's connection between CON1 and CON2. Otherwise, their gates are pulled up to their sources by the 220kΩ resistor and they switch off, opening the Switch Module's connection between CON1 and CON2.

Toggling the Switch Module state involves pulling either of Q1 or Q2's gates to LOGIC_GND. This can be



The I/O Expander (also labelled as an I^{2}C Interface) adds another PCB to the Battery Manager stack. If you need multiple I/O Expanders, you could fit them with stackable headers (as used on Arduino Shields). Just be sure to set different I^{2}C addresses on the stacked PCBs.

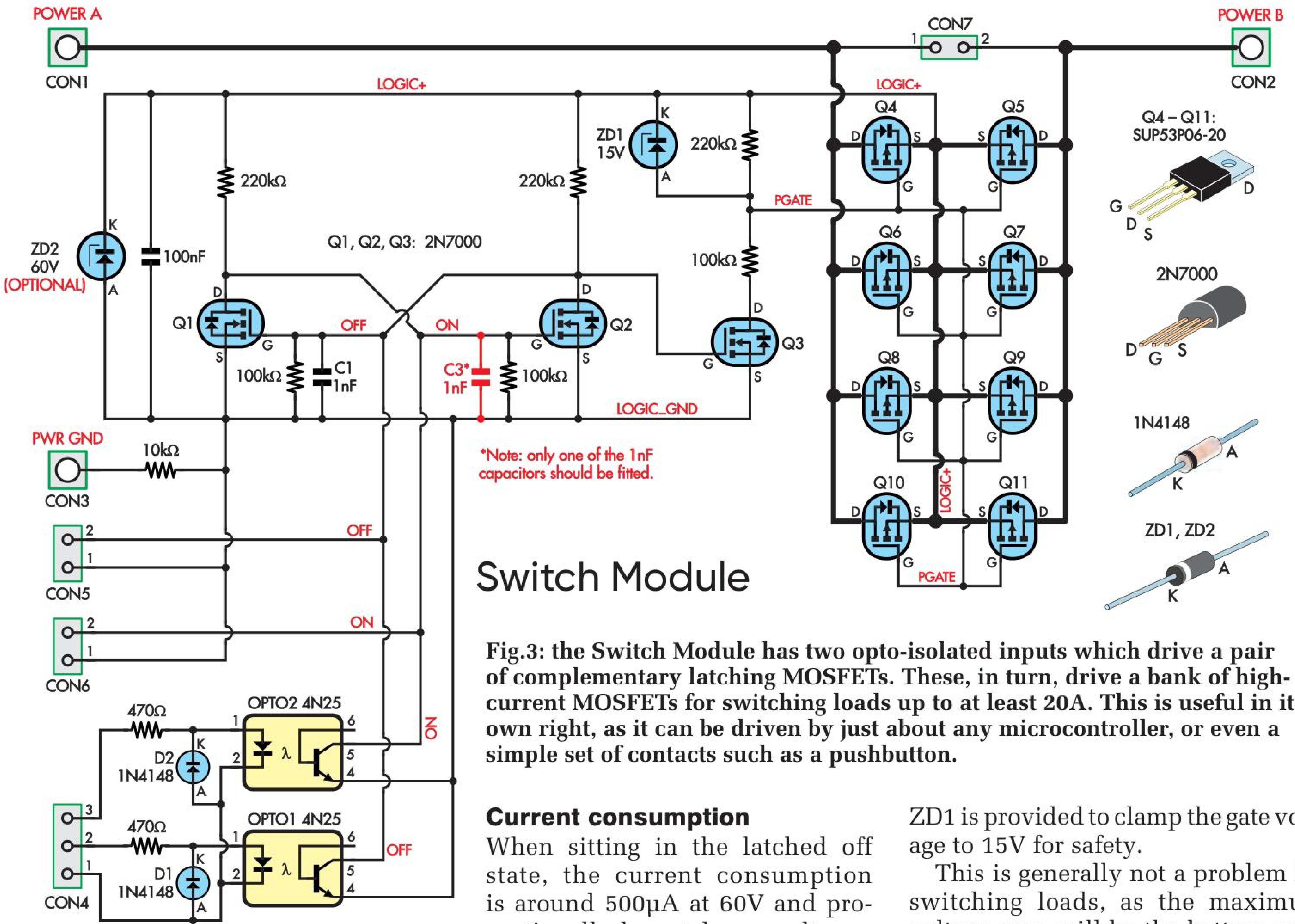


Fig.3: the Switch Module has two opto-isolated inputs which drive a pair of complementary latching MOSFETs. These, in turn, drive a bank of high-current MOSFETs for switching loads up to at least 20A. This is useful in its own right, as it can be driven by just about any microcontroller, or even a simple set of contacts such as a pushbutton.

done by the phototransistor outputs of OPTO1 or OPTO2, respectively.

When a voltage is applied at CON4's pin 2 that is positive with respect to its pin 1, current flows through OPTO1's LED via the 470Ω resistor, turning on its phototransistor. Similarly, a positive voltage at pin 3 of CON4 triggers OPTO2, pulling Q2's gate low.

A connection between the pins of CON5 or CON6 will have the same effect. This allows control by something like a pushbutton, in addition to control by the *Battery Manager*.

If both Q1 and Q2 have their gates pulled low, then naturally, Q3's gate is low too, and the Switch Module is off. Thus the safe 'off' state dominates if conflicting signals occur.

This is similar to the state that occurs when power is first applied. In this case, capacitor C1 (which will have been discharged by its parallel 100kΩ resistor) holds Q1's gate low for a brief period, allowing Q2 to turn on before Q1, and the Switch Module is forced into the off state.

The time constant of this RC network is less than 1ms, so as long as external pulses are at least this long, then incoming pulses are latched correctly. If the reverse behaviour is required, then the capacitor is fitted adjacent to Q2, to the pads marked C3 instead of C1. The Switch Module will then power up in the on state.

Current consumption

When sitting in the latched off state, the current consumption is around 500μA at 60V and proportionally less at lower voltages; around 100μA at 12V. When switched on, extra current flows through Q3, adding around 200μA at 60V, down to 40μA at 12V.

The current level during switching will be higher than this, with Q1, Q2 and Q3 sinking current, but that occurs only very briefly, because the complementary transistors turn off almost instantaneously.

Switch ratings

P-channel MOSFETs Q4-Q11 are SUP53P06 types with a nominal maximum gate voltage of –20V (with respect to the source) and a maximum drain voltage of –60V. These parameters set the practical working limits of the Switch Module.

These MOSFETs are specified at around 9A continuous current each (at 25°C), but the PCB track width limits this to about 20A across the four pairs; perhaps up to 30A with ample cooling. This can be increased by supplementing the PCB with extra wires soldered directly to the MOSFETs.

Alternatively, for very light loads, some MOSFETs could be left off.

The dividers around Q1, Q2 and Q3 have been set to allow operation up to 60V (respecting their 20V gate limit with the 220kΩ/100kΩ divider). Since they have gate thresholds down around 3V, they require a battery voltage of at least 10V to work correctly. ZD1 is provided to clamp the gate voltage to 15V for safety.

This is generally not a problem for switching loads, as the maximum voltage seen will be the battery voltage. However, for charging sources the voltage can be much higher. For example, a 12V solar panel can have a 22V open-circuit voltage. Wind turbines can be even higher; they typically need a shunt regulator to prevent their unloaded voltage from reaching dangerous levels.

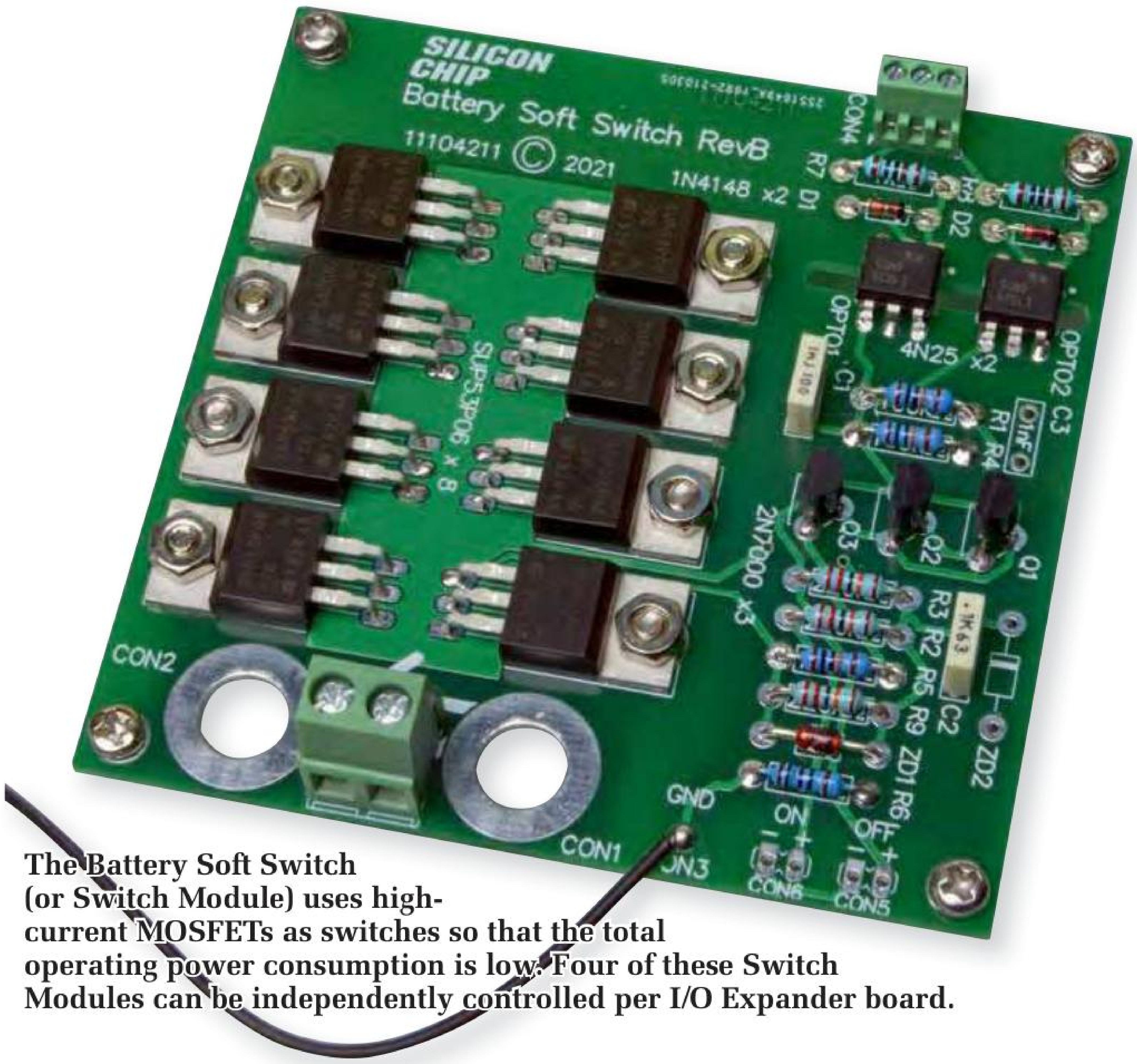
So care should be taken when using this module with a charging source to ensure that the open-circuit voltage does not exceed the Switch Module's limits.

The MOSFET types can be changed to allow operation at higher voltages, but other parts of the circuit might have to be modified too. For example, the SPP15P10-type MOSFET used in the *Burp Charger for NiMH and NiCad Batteries* (*PE*, June 2015) can handle up to 100V, and is a direct substitute for the SUP53P06.

The other change we recommend for higher voltage builds is increasing the value of the 10kΩ resistor to reduce the quiescent current through it and possibly ZD2.

Consider the section in parallel with ZD2 as having a resistance of around 100kΩ. So for 100V switching, replacing the 10kΩ resistor with a 220kΩ resistor will put around 30V across ZD2, thereby allowing the circuit to operate correctly.

We have not specified the SPP15P10 MOSFET in our parts list because it



The Battery Soft Switch (or Switch Module) uses high-current MOSFETs as switches so that the total operating power consumption is low. Four of these Switch Modules can be independently controlled per I/O Expander board.

has a much higher drain-source resistance. So it will produce more heat at the same current level, and we expect most readers will be using the Switch Module below 60V.

Handling more current

If you find that your current requirements are beyond that of the Switch Module, you can use the Switch Module to operate the coil of a heavy-duty relay. The current when energised will be much higher, but this option allows the *Battery Manager* to work with just about any load.

In this case, just a single pair of MOSFETs is sufficient to operate the relay coil (eg, Q4 and Q5). A snubbing diode across the coil is highly recommended, to protect the MOSFETs from spikes that the coil might generate when it de-energises.

Software updates

This is a good point to upgrade the software on the *Battery Multi Logger* to give it the *Battery Manager* features, if for no other reason than to get it out of the way before we connect the I/O Expander to the ICSP header (which would make programming trickier).

If you have blank chips, follow the original instructions for programming the *Battery Multi Logger*, including putting the Microbridge firmware on the PIC16F1455. But instead of the *Battery Multi Logger* firmware file (**1110620A.HEX**), load the newer *Battery Manager* (**1110620B.HEX**) file.

Don't forget to set JP2 to the PROG position before using the ICSP interface – and then set it back to RUN after programming.

For an already-working *Battery Multi Logger*, you can simply update the MMBasic file. The same library file and LCD OPTIONS are used, so no other changes are needed. The act of loading a program will delete any logged data, so you should export that first, if necessary.

The newer software has less space for logged data due to needing more space to store configuration variables for the Soft Switches. Thus, the long-term data is reduced to 10 days, and this allows two Soft Switches (controlled by four triggers) to be fitted.

These limits are set by CONST values in the program. We've listed some options below regarding how these two values can be changed and still fit within the existing Flash memory. But generally, as long as the sum of the number of days stored and the number of triggers is no more than 14, it should work.

These are the D_COUNT and TRIG_COUNT values. Due to the way they are displayed on the page, TRIG_COUNT should be no higher than eight, as otherwise, the control buttons cannot be seen.

You will need to load the 'crunched' (with comments and whitespace removed) program, as it does not fit in memory otherwise. The uncrunched version is also available so that you can inspect the fully-commented code and make changes if you like.

In general, you should follow the instructions for the *Battery Multi Logger* but replace the respective HEX and MMBasic files with their *Battery Manager* equivalents.

Run the newly installed program to set the AUTORUN flag. Now it should automatically start when powered up. We'll go into the software detail later, but you should see the new main page as seen in Screen 1.

Construction options

An important first step before building the board is to determine what parts are needed. Given the low cost of the parts for the I/O Expander board, we recommend that you build the full version, which can handle four Switch Modules.

However, you could leave off some of the parts if you are sure that you will only be connecting one or two Switch Modules.

The specifics of your battery installation might also affect how you build it. We've designed the Switch Module PCB with holes to suit 8mm screws and thus eyelets suiting up to 8G (3.25mm diameter copper) cable, which should be sufficient for anything that the Switch Module can handle.

You will need to consider how many Switch Modules you need. Most people will need one or perhaps two to disable non-critical loads when the battery charge state gets low.

There's always a critical load that can't be disconnected, and that won't need a Switch Module; you'd much rather have a flat battery than a submerged boat because the bilge pump wasn't running!

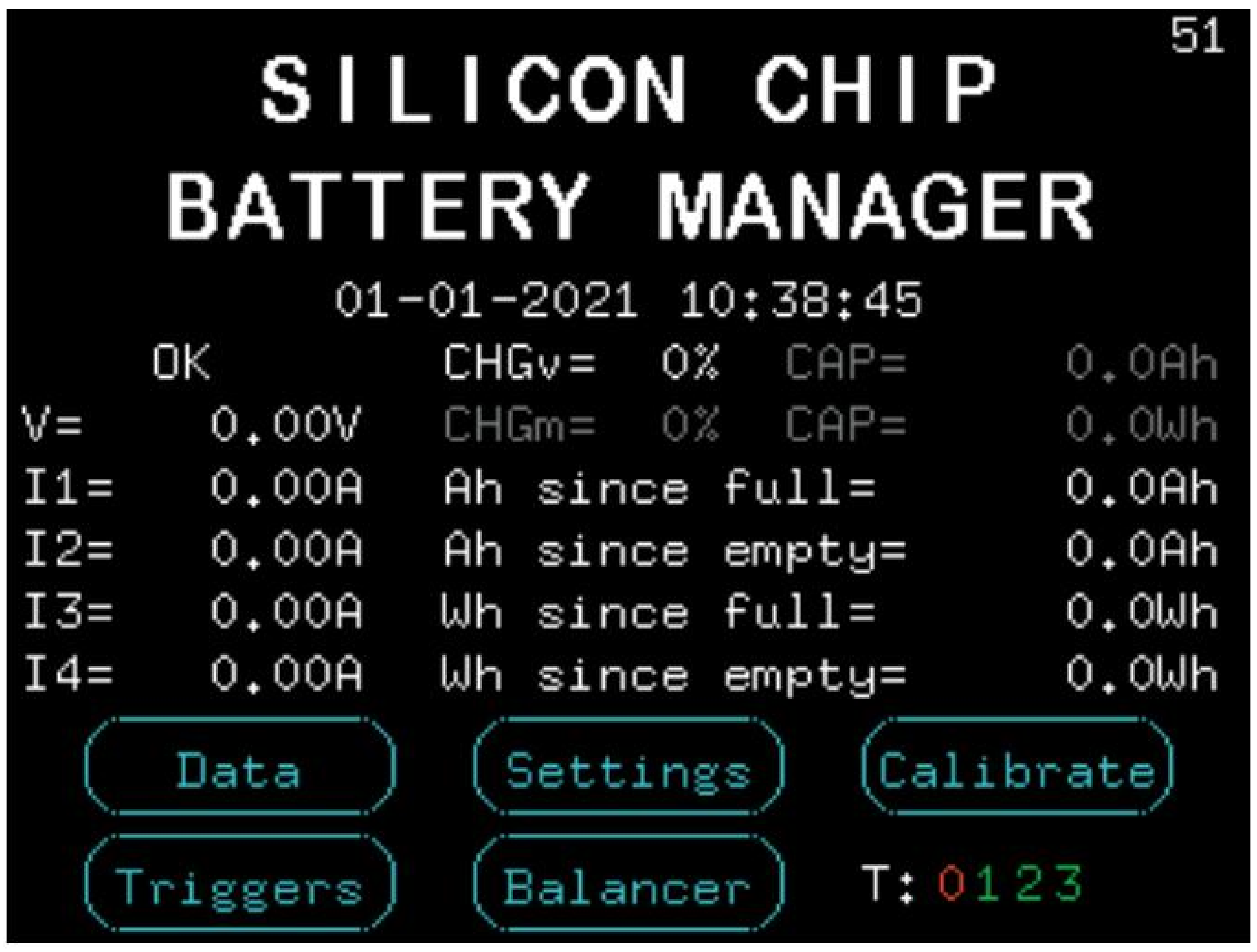
And assuming you have a reputable charge controller, there will be little need to add a Switch Module inline with any connected solar panels.

Similarly, you might or might not need to build and connect a *Battery Balancer*. If you have a 24V, 36V or 48V system composed of 12V batteries wired in series, you can make good use of the *Battery Balancer*. If the *Battery Balancer* and *Battery Multi Logger* can't share the same ground, you will also need to build the *Mini Isolated Serial Link*.

Regardless, we recommend using the *Mini Isolated Serial Link* to avoid any potential problems; it's cheap and easy to build, and safer to isolate the two devices.

Building the I/O Expander

Let's start by building the I/O Expander and connecting it to the *Battery Multi Logger*; you can add switch Modules after that. If you're just interested in the *Battery-Balancer*-related upgrades,



```
                                              51
              SILICON CHIP
           BATTERY MANAGER
           01-01-2021 10:38:45
     OK          CHGv=  0%   CAP=        0.0Ah
V=     0.00V     CHGm=  0%   CAP=        0.0Wh
I1=    0.00A     Ah since full=          0.0Ah
I2=    0.00A     Ah since empty=         0.0Ah
I3=    0.00A     Wh since full=          0.0Wh
I4=    0.00A     Wh since empty=         0.0Wh
```

Screen 1: the Battery Multi Logger's Main screen has been updated to add two new buttons for the Trigger and Balancer functions.

At bottom right, the trigger state is shown, and the title has also been changed to reflect the unit's new capabilities.

you can skip most of the construction (assuming you've already built the *Battery Multi Logger* and *Battery Balancer*).

The I/O Expander is built on a double-sided PCB coded 11104212, which measures 37.5 x 35.5mm and is available from the *PE PCB Service*. Fig.4 is the PCB overlay diagram; all the components are on one side, but there are some solder pad 'jumpers' on the underside, so both sides are shown.

Since the I/O Expander will essentially become part of the *Battery Multi Logger* PCB, we have used surface-mounted parts. We recommend having on hand a fine-tipped soldering iron, flux paste, solder wicking braid, a set of tweezers and a magnifier. Flux paste releases a fair bit of smoke, so good fume extraction or ventilation is important too.

Start by fitting IC1, noting that its pin 1 is closest to the mounting hole. We found the marking on this chip difficult to discern; there should be a small circle on the top of the part, and a bevel along the nearest edge. On the chips we used, pin 1 is at lower left when the chip markings are the right way up (with the bevel along the bottom edge).

Put some flux on the PCB pads for IC1, rest the chip roughly in place and apply a bit more flux to the top of the pins. It doesn't hurt to be generous.

Load the tip of the iron with a bit of solder and tack one pin in place. Adjust the chip if necessary by melting the solder and nudging the chip with tweezers. Once all the pins are correctly aligned, solder them to the PCB.

If there are any bridges between pins, remove them with the braid. Add flux to the bridge and press the braid against it with the iron, carefully pulling it away when the braid has drawn up the solder.

Solder the transistors next. They are all the same type and are polarised, but should only fit one way due to their shape. Put some flux on the pads and tack one lead in place, then solder the remaining leads. Despite their small size, the leads are well spread around the part, so they are quite easy to solder.

Place the three capacitors next; they are near the top of the PCB. Use a similar technique of soldering one lead at a time. The remaining parts are much easier to solder and have larger pads.

Follow with the resistors, then clean up any excess flux with the solution recommended by the flux manufacturer (or your favourite one).

Attaching it to the main board

Since the I/O Expander is designed to mount directly to the *Battery Multi Logger* PCB, shut down the *Logger* and disassemble it. If you have a header fitted to CON2, remove it and clean up the pads to allow the I/O Expander to be fitted.

Take the pairs of header pins and sockets and plug them together. Install them in their respective holes between the two PCBs, with the female headers on the *Logger* PCB and the male headers on the I/O Expander PCB. This will reduce the chance of exposed connectors if the I/O Expander PCB is removed.

You can then clamp the two PCBs together temporarily with a machine screw and nut (or tapped spacer). This will make them easier to solder. Refer to the photos as a guide.

Solder the headers in place, remove the temporary screw and reassemble the stack, including the LCD. Instead of fitting the machine screw in the corner where the I/O Expander sits, use the extra spacer and the longer machine screw to secure everything against the tapped spacer fitted to the back of the LCD.

Switch Module assembly

The Switch Module is built on a PCB coded 11104211, which measures 81.5 x 82.5mm and is available from the *PE PCB Service*. Its overlay diagram is shown in Fig.5 – note that it uses all through-hole parts. Start by fitting the resistors according to the markings on the PCB. It's best to check their values with a multimeter to ensure you have the correct components.

Follow with the two zener diodes. Neither of these are necessary for systems that operate up to around 25V, as there are unlikely to be voltages high enough to cause damage to MOSFETs, although it's a good idea to fit ZD1 to protect the MOSFETs. ZD2 is only needed for systems that go over 60V.

Keep in mind what we mentioned before about solar panels and wind turbines producing much higher voltages than their nominal ratings.

Next, fit the two 1N4148 diodes near CON4, noting their polarity.

Follow with the two adjacent opto-isolators. Take care that their pin 1 markings align as shown in our photos. They both face the same way.

Now install the two capacitors. As mentioned earlier, C3 does not need to be fitted unless the default behaviour needs to be changed, so it is not shown in Fig.5.

After this, mount the three smaller transistors, Q1-Q3. Ensure that they align to their footprints, and push them down as close to the PCB as possible before soldering.

Follow with the larger transistors. If you are not fitting all of them, fit those closest to CON1 and CON2 in

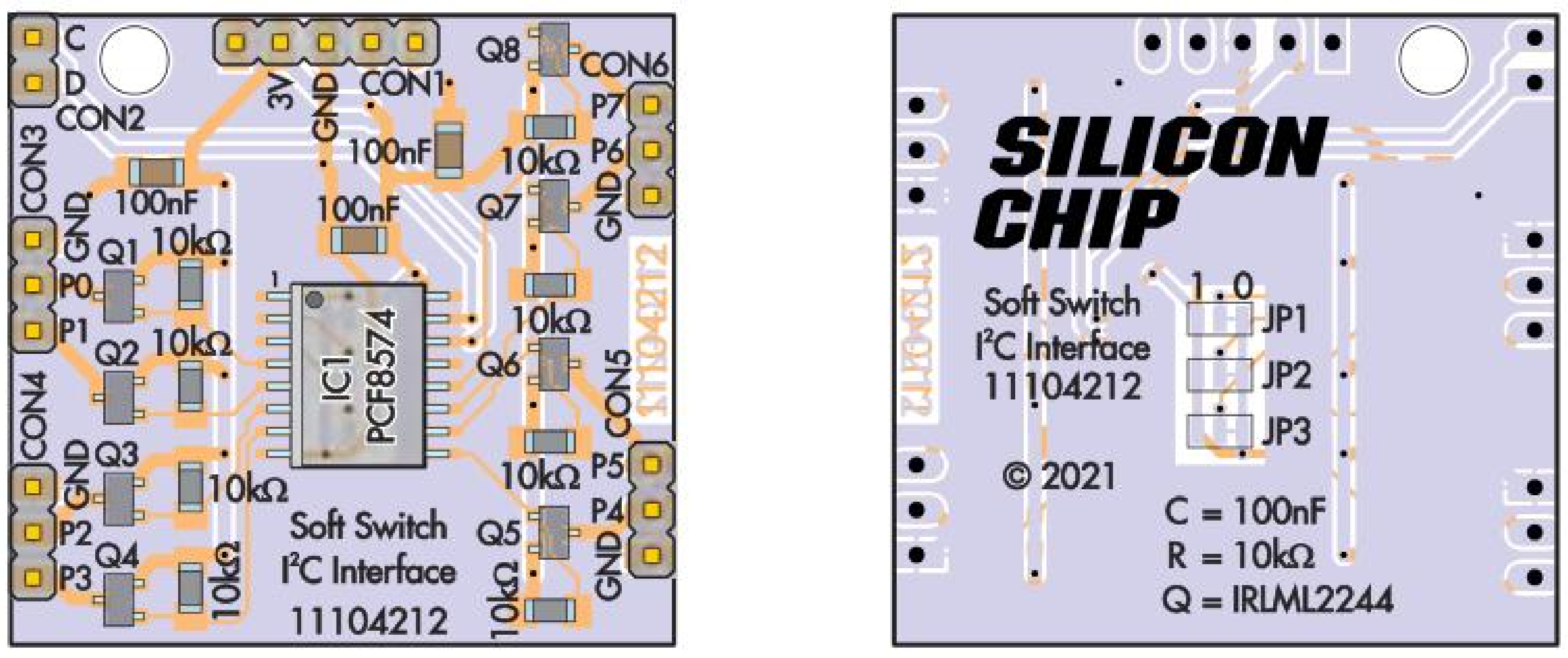


Fig.4: assembly of the I/O Expander is straightforward – it uses mostly SMD components, but they are easy to handle. Fit IC1 first, ensuring its pin 1 marking is oriented as shown. All capacitors and resistors are non-polarised and of a single value. You don't have to use vertical headers, as shown here; you could use right-angle headers, sockets or just solder wires to the pads.

matching pairs. For example, if you only need four MOSFETs to handle your load current, put them in the spots marked Q4-Q7.

For each transistor, bend its leads back 90° around 7mm from where they meet the body. Insert the leads through the PCB and fix the tab in place with the machine screw, washer and nut. Take care not to twist the transistors, which might bend the leads.

Once aligned with its footprint, solder and trim the leads. The large copper pour will draw heat from your iron, so use a higher temperature if necessary. We've added some thermal relief on the PCB to help with this.

Basic testing

You might like to test the I/O Expander and Switch Module at this point. Connect CON3 on the I/O Expander to CON4 on the Switch Module. Connect G to COM, P0 to OFF and P1 to ON.

Now attach a 12V power source between CON1 and CON3 on the Switch Module, with the negative terminal to CON3. Connect a multimeter across the empty C3 pads; it should read about 1/3 of the supply voltage.

Shorting the CON6 pads on the Switch Module should cause this to drop to 0V and stay there when released. Similarly, shorting CON5's pads will cause the voltage to revert to 1/3 supply.

Using the SOFT SWITCH page on the *Battery Manager*, you can press the green button next to TR0 and TR1 to toggle the state via the I/O Expander. Keep in mind that the software has been configured with some defaults to suit a 12V battery, and these will be active when the *Battery Manager* is first powered up.

If all this is correct, then the I/O Expander and Switch Module are working correctly. Your wiring from here will depend on your application, but consider that CON1 and CON2 are the switch terminals.

Ideally, you should have a fuse and separate switch to the battery circuit feeding the Switch Module to protect it in the event of a fault.

So, do ensure that you don't connect something that can cause damage or be affected by unplanned switching. You might like to leave this until later, after you have configured the *Battery Manager*.

Note the holes in the corner of the PCB, which are designed to take M3 machine screws, allowing the Soft Switch modules to be mounted in an enclosure. For example, you could fit them to the interior of the same panel as the *Battery Manager*.

Parts List – Battery Manager

1 assembled Battery Multi Logger module
(*PE*, February-March 2022, with IC1 programmed with 1110620**B**.hex instead of 1110620**A**.hex
1 assembled Battery Balancer module (**optional**) (PE, April-May 2022)
1 or more assembled I/O Expander modules (see below)
1 or more assembled Switch Modules (see below)
1 assembled Mini Isolated Serial Link (**optional**) (PE, March 2022)
various lengths of heavy-duty wire, eyelet lugs etc to suit battery and application
various lengths of medium-duty hookup wire (see text)
various jumper leads (**optional**; to connect I/O Expander module[s] to the Battery Manager and Switch Module[s])

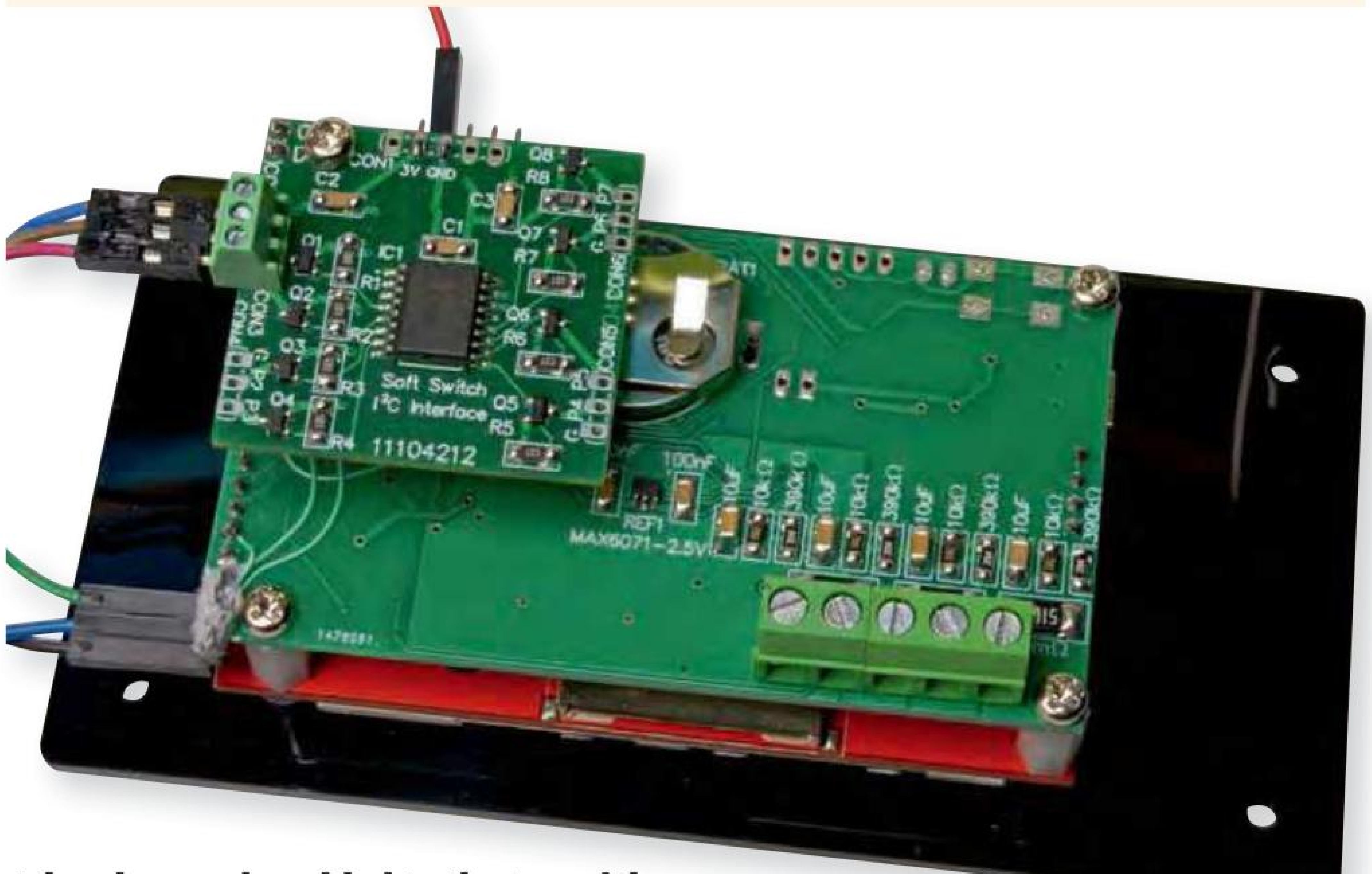
I/O Expander module parts (per module)
1 double-sided PCB coded 11104212, 38 x 36mm
1 PCF8574 I^2C expander IC, SOIC-16 (IC1) **[Digi-Key, Mouser]**
8 IRLML2244 P-channel MOSFETs (Q1-Q8) **[Digi-Key, Mouser]**
3 100nF X7R SMD 3216/M1206-size ceramic capacitors
8 10kΩ 1% SMD 3216/M1206-size resistors
1 5-way header (CON1)
1 2-way header (CON2)
1-4 3-way headers or subminiature screw terminals� (CON3-CON6)
1 untapped 12mm-long spacer, ~3.125mm inner diameter
1 M3 x 20mm panhead machine screw

Switch Module (per module)
1 double-sided PCB coded 11104211, 82 x 83mm
1 3-way pin header or subminiature screw terminal� (CON4)
16 M3 x 6mm panhead machine screws
4 M3 x 12mm tapped spacers
8 M3 nuts
8 M3 washers
� *eg, Digi-Key part number ED10562*

Semiconductors
2 4N25 opto-isolators, DIP-6 (OPTO1,OPTO2)
3 2N7000 N-channel small-signal MOSFETs, TO-92 (Q1-Q3)
2-8 SUP53P06 P-channel high-current logic-level MOSFETs, TO-220 (Q4-Q11)
1 15V zener diode (ZD1)
1 39-60V zener diode (ZD2) (**optional**; see text)
2 1N4148 small signal diodes (D1,D2)

Capacitors
1 100nF 100V MKT
1 1nF 100V MKT

Resistors (all 1/4W 1% axial metal film)
3 220kΩ
3 100kΩ
1 10kΩ
2 470Ω



A header can be added to the top of the Logger PCB, as shown, to allow in-circuit programming. This header can also provide power to a Mini Isolated Serial Link for connection to a Battery Balancer (singular red wire).

Battery Balancer interface

You need four wires to connect the *Battery Balancer* to the *Battery Manager* if using the *Mini Isolated Serial Link*, or three if you are not. The fourth wire is to power the isolator.

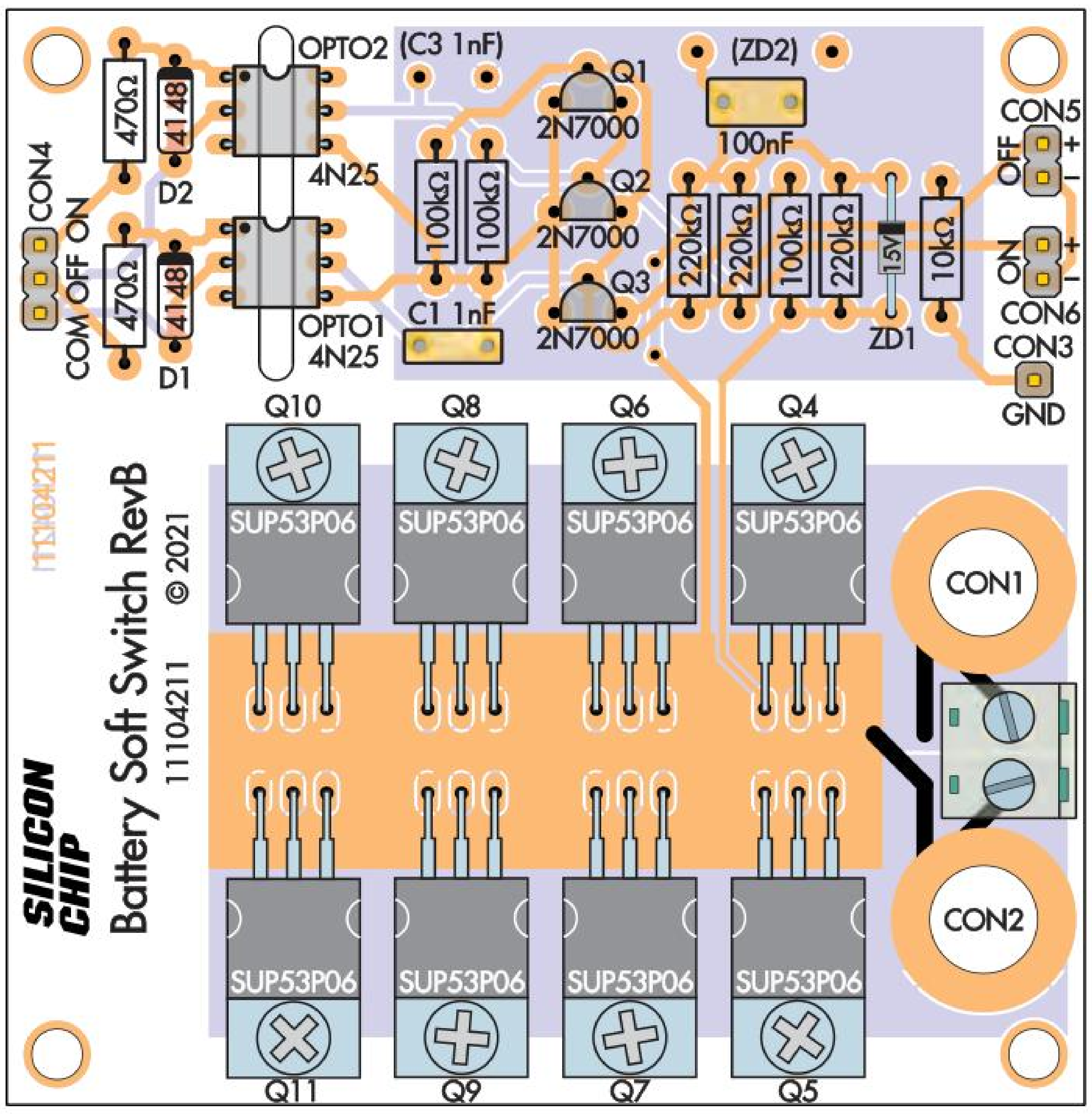

OPTO2
(C3 1nF)
(ZD2)
Q1
2N7000
100nF
Q2
Q3
4N25
CON4
COM OFF ON
D2
D1
470Ω
4148
100kΩ
220kΩ
10kΩ
15V
ZD1
C1 1nF
OPTO1
CON5
OFF
ON
CON6
CON3
GND
Q10
Q8
Q6
Q4
SUP53P06
CON1
CON2
Q11
Q9
Q7
Q5
SILICON CHIP
Battery Soft Switch RevB
11104211
© 2021

Fig.5: the Switch Module uses all through-hole components and is easy to assemble. Watch the orientation of OPTO1, OPTO2 and the diodes. You can install fewer than eight MOSFETs if your load draws less than 20A; just make sure to fit them in pairs (Q4 and Q5; Q6 and Q7...). The load can be connected either via the two-way screw terminal, or eye lugs bolted to CON1 and CON2.

Revision E and later of the *Battery Multi Logger* PCB has pads breaking out the three connections at CON6. For power, you will need to tap into the 3.3V supply, and the best place for this will be at the *Battery Manager's* CON2 (which also connects to the I/O Expander's CON1).

If you have an earlier PCB (unlikely, but see note on caption to Fig.6), then the only way to tap into the serial data pins (Micromite pins 21 and 22) is to solder directly to the pins at the IC. It's not easy, but it is not much harder than soldering the SOIC parts in the first place.

Fig.6 and 7 show the wiring required. Fig.6 depicts how a direct connection would be made, while Fig.7 shows the wiring via a *Mini Isolated Serial Link*.

Note how in both cases, the wires appear to go to two points on the *Battery Logger* PCB at left. They only need to go to one. If CON6 is present (on Revision E boards or later), then use those connections. Otherwise, use the dashed alternatives. These go to pin 22 of the IC for RX and pin 21 for TX.

If CON6 is missing, the ground connection can be taken from pin 2 of the LCD header or the middle pin (pin 3) of CON2, the ICSP header.

The preferred arrangement, using the *Mini Isolated Serial Link*, is shown

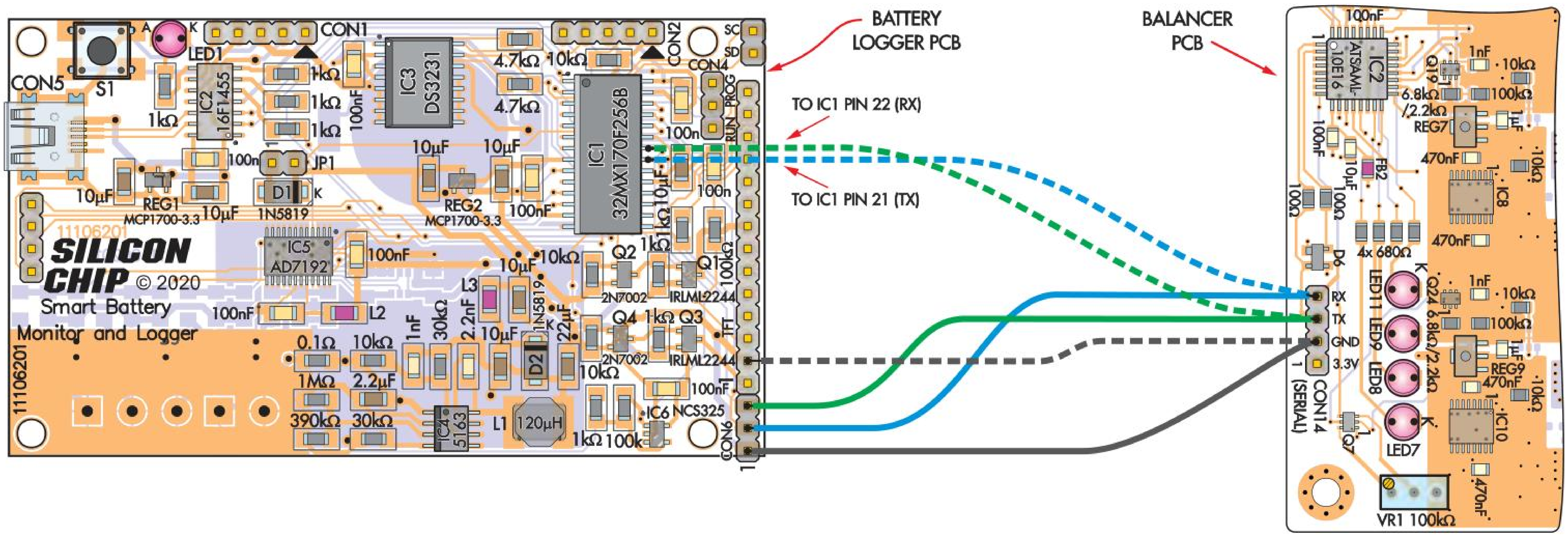

BATTERY LOGGER PCB
BALANCER PCB
TO IC1 PIN 22 (RX)
TO IC1 PIN 21 (TX)
IC1 32MX170F256B
IC3 DS3231
IC5 AD7192
CON6
SILICON CHIP © 2020
Smart Battery Monitor and Logger
RX
TX
GND
3.3V
CON14 (SERIAL)
IC2 ATSAML10E16
VR1 100kΩ

Fig.6: only one of each colour of wire is needed, but we've shown two options for each, so you can choose a suitable way to connect the two boards. The dashed wires are only needed if you have an early revision of the PCB that lacks CON6 (bottom right). While the boards are notionally at the same ground potential, it wouldn't hurt to add series resistors, but Fig.7 shows an even better option. (Note – all boards sold by PE will be of the later variety with CON6.)

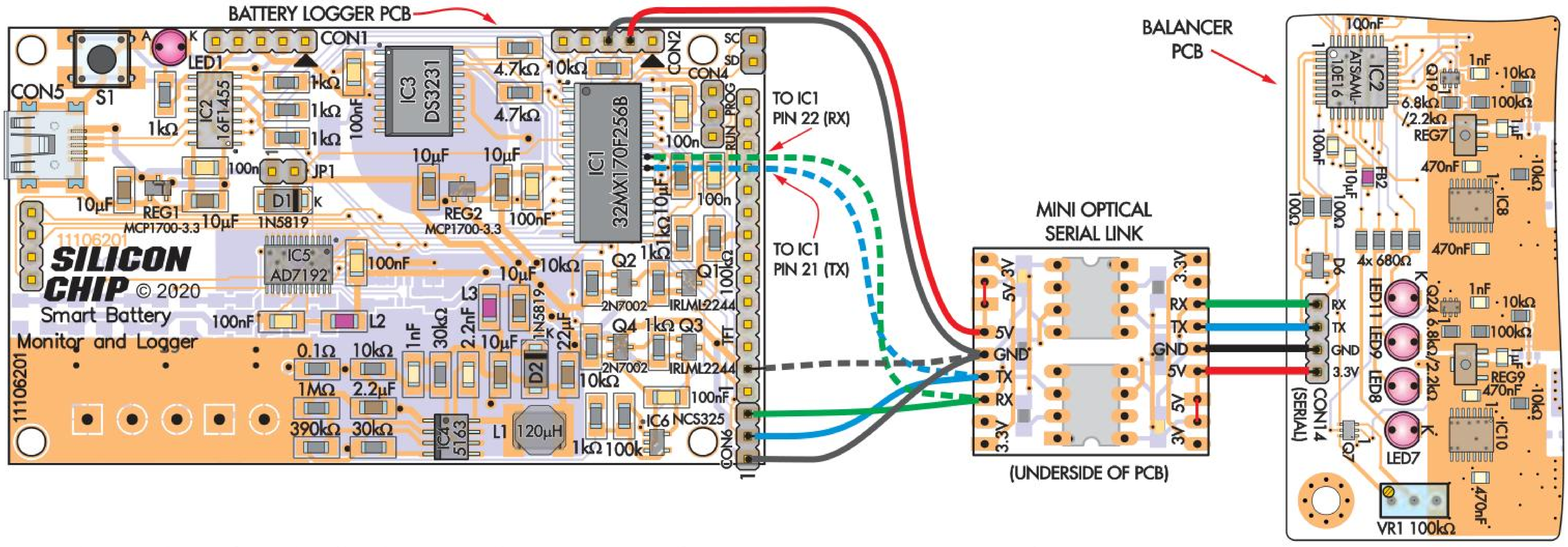

BATTERY LOGGER PCB
BALANCER PCB
TO IC1 PIN 22 (RX)
TO IC1 PIN 21 (TX)
MINI OPTICAL SERIAL LINK
5V
3.3V
GND
TX
RX
(UNDERSIDE OF PCB)
CON14 (SERIAL)
VR1 100kΩ

Fig.7: the preferred method of joining the Battery Multi Logger to our Battery Balancer is via a Mini Isolated Serial Link module. The module needs to be supplied with 3.3V on each side; ensure that the jumper links on the Serial Link are set to the 5V positions, as shown (which actually corresponds to 3.3V in this case).

in Fig.7. Jumpers JP1 and JP2 on the *Isolator* board are set to the 5V position, which means it takes power from the pin adjacent to ground.

Since the *Battery Balancer* has been designed to have the *Mini Isolated Serial Link* directly attached, it makes sense to do this, as it matches that configuration. Then run the four wires back to the *Battery Multi Logger* PCB.

If the *Mini Isolated Serial Link* is fitted upside-down to the *Battery Balancer* PCB (as in Fig.7), it will not hide the LEDs, although it will slightly overhang the PCB edge. The photo on page 71 shows the Link fitted to the *Balancer* in this fashion.

Due to space constraints, there is no 3.3V connection on CON6, so the best option is to take this from pin 2 of the ICSP header. If you lack CON6, then taking the ground connection from the adjacent pin 3 is a good choice.

Similarly, the TX and RX signals are taken from CON6 or the microcontroller pins directly, as shown.

While setting up these connections, you might also like to solder a five-way header to either CON2 of the *Battery Logger* or CON1 of the I/O Expander to regain the in-circuit programming (ICSP) capability. All the things we have hanging from these pins only take power and ground connections – therefore, they should not affect programming.

But you may have to power the board from USB instead of the programmer during ICSP programming, as the programmer might not be able to provide sufficient current.

Reassemble anything you have taken apart during this construction. Then power up the *Battery Logger* and its connected peripherals.

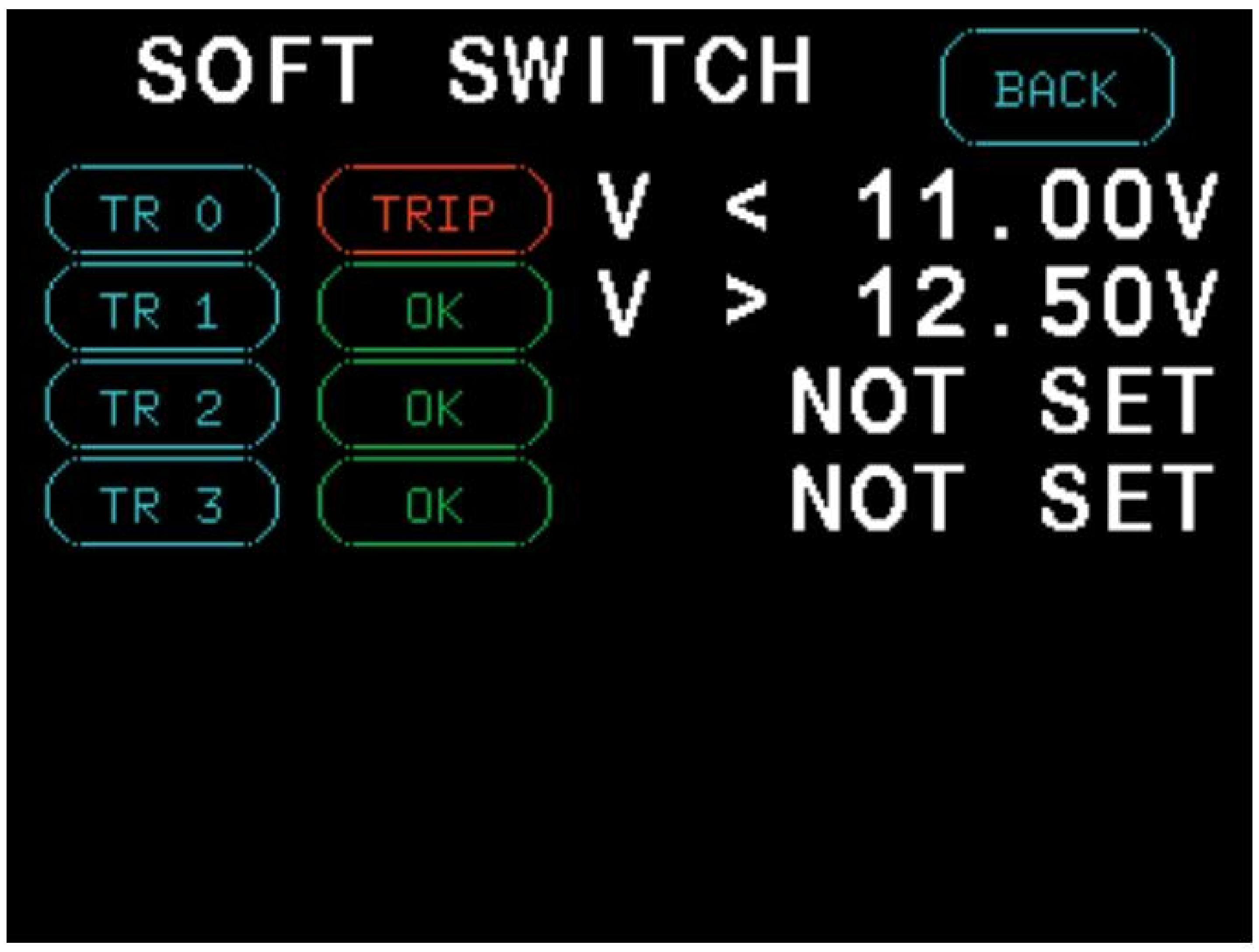
Using it

Now that everything has been configured, we can explore the new screens. Screen 1 is the updated Main screen, with two new buttons and a display for the status of the triggers.

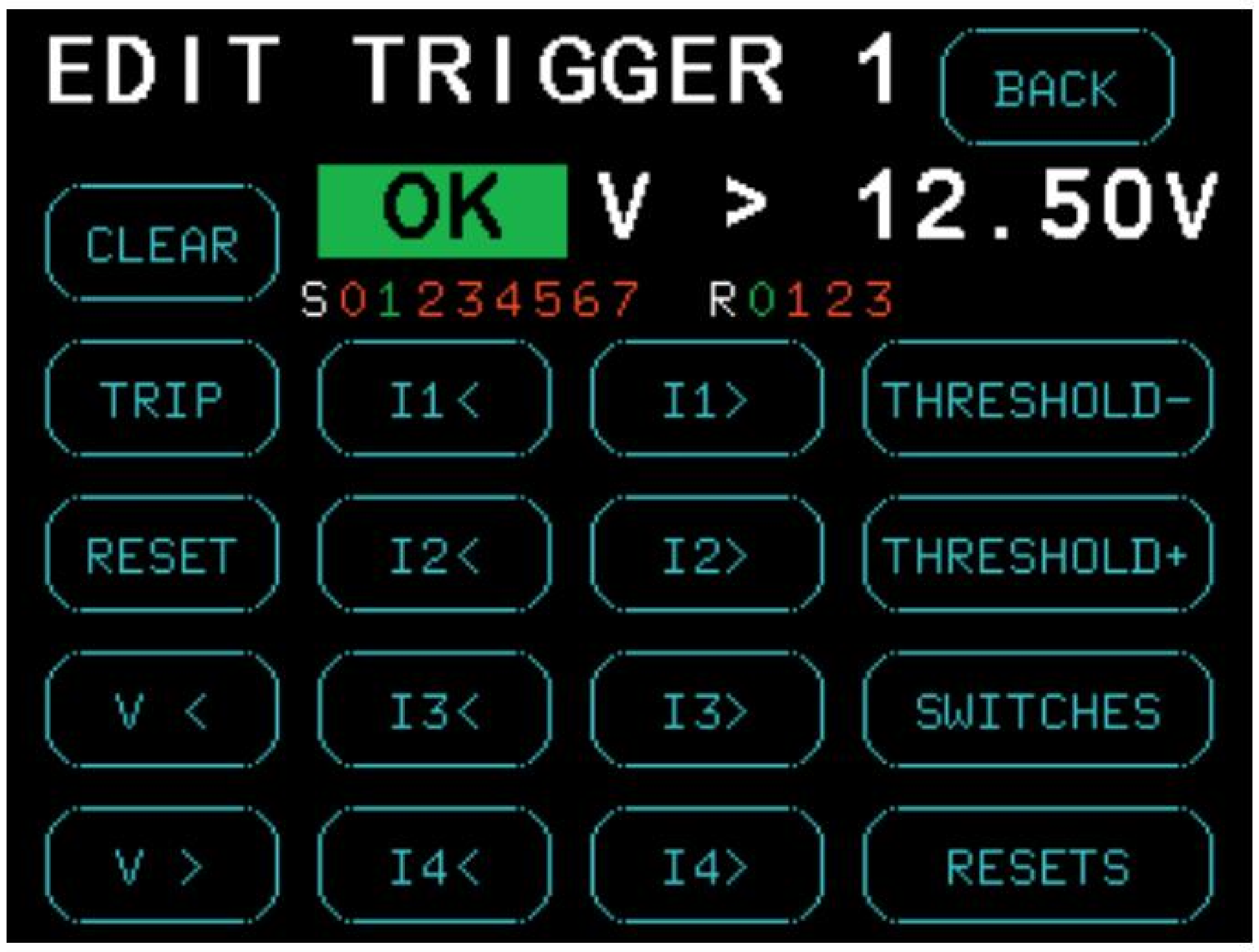
If your battery is above 12.5V, you should see Trigger 1 in red. Or if your battery is below 11V, then Trigger 0 might have tripped.

Press the Trigger button to see Screen 2. This is an overview of the triggers, with one displayed on each line. Each trigger has a parameter and threshold that it monitors; these are displayed as in Screen 2. When a parameter reaches its threshold, the trigger is tripped and will show a red TRIP button instead of a green OK button. The trigger cannot trip again until it is reset.

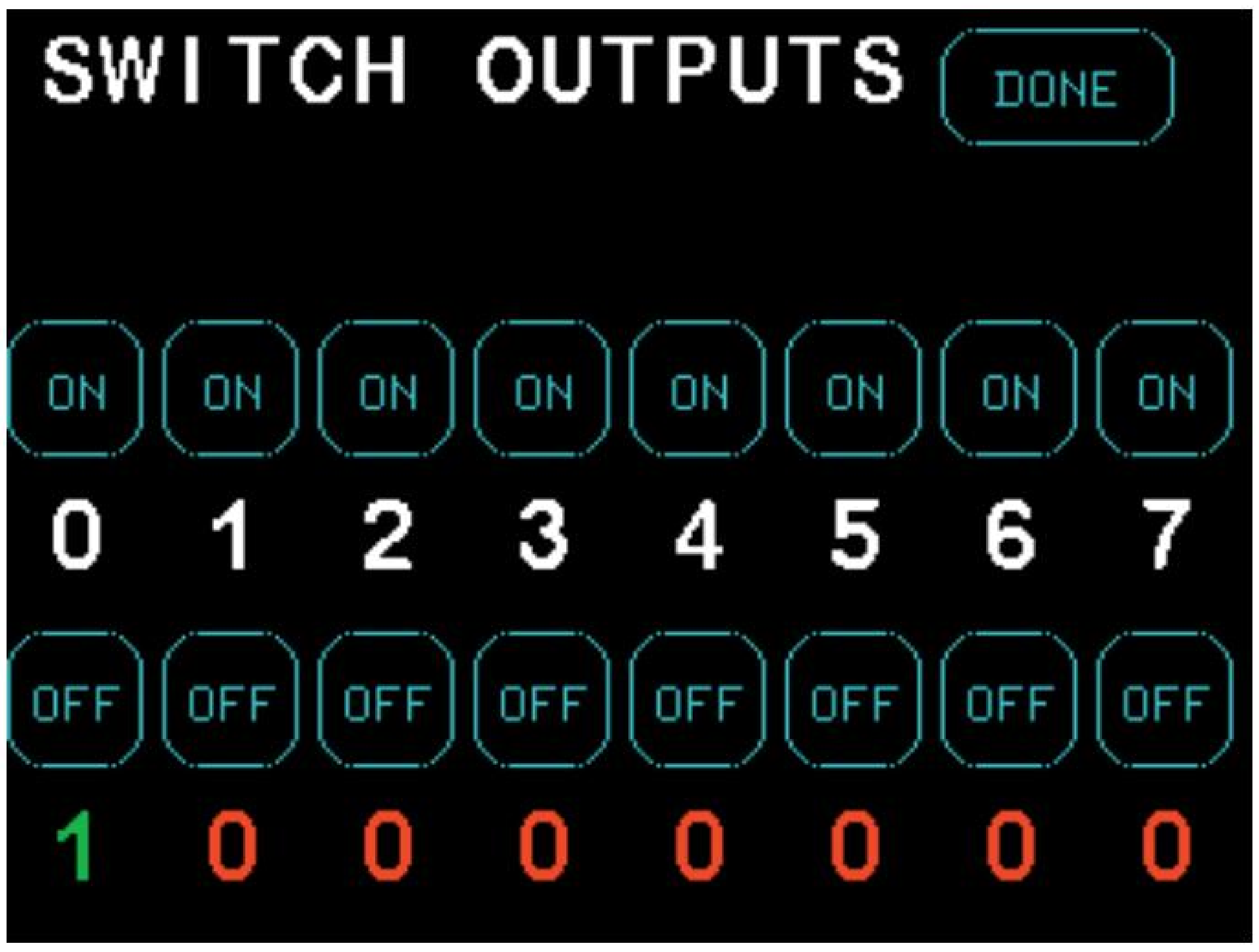
On each trip event, any combination of switches can be activated. These switches correspond to Soft Switch inputs, and the software delivers a pulse via the I/O Expander to the corresponding switches.



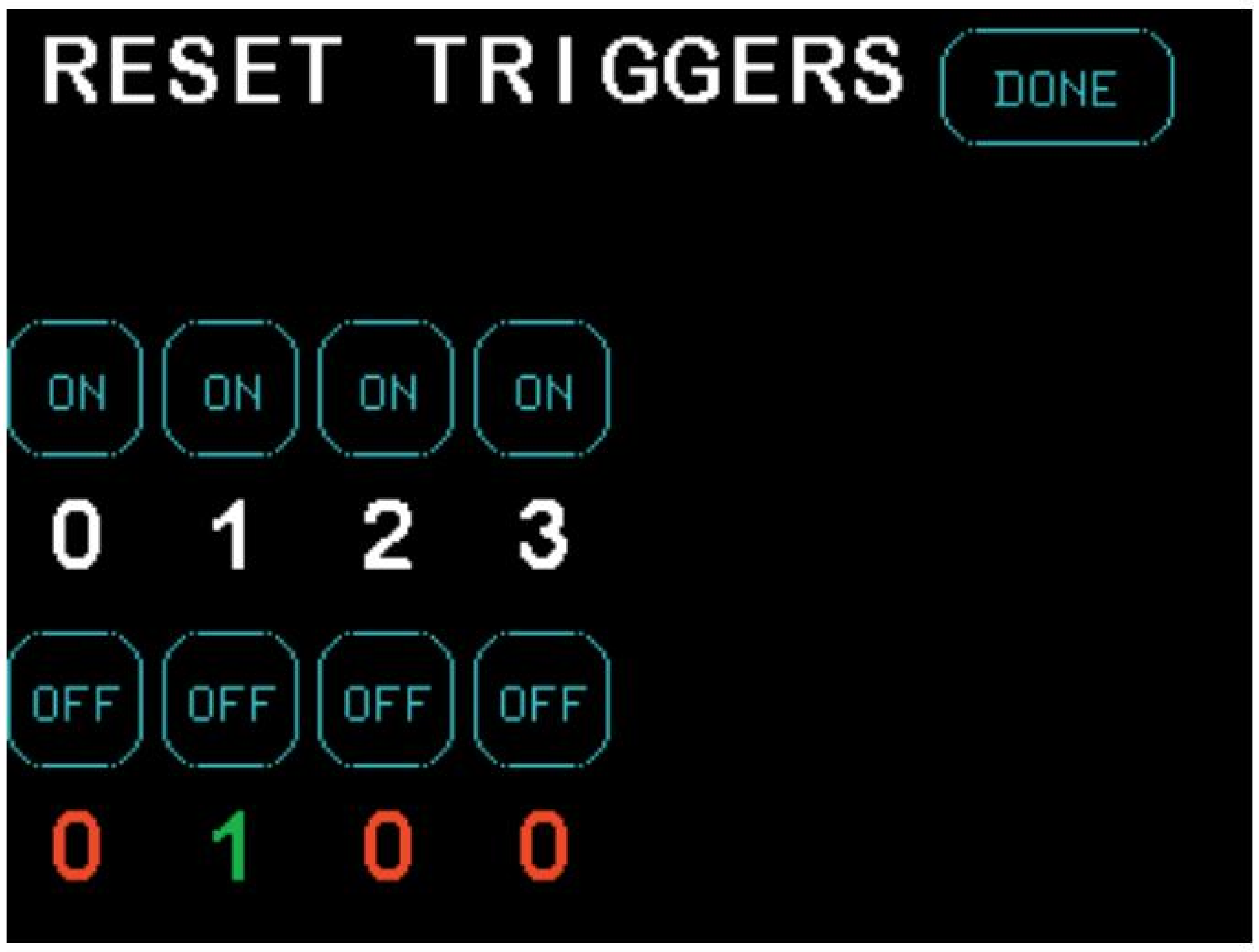
Screen 2: the SOFT SWITCH page shows the trigger states and thresholds. Pressing the buttons allows the triggers' operation to be tested and triggers to be manually reset, if this form of operation is preferred.



Screen 3: each trigger is configured on its EDIT TRIGGER page, including its thresholds. The page displays the switches it drives and the other triggers it will reset.

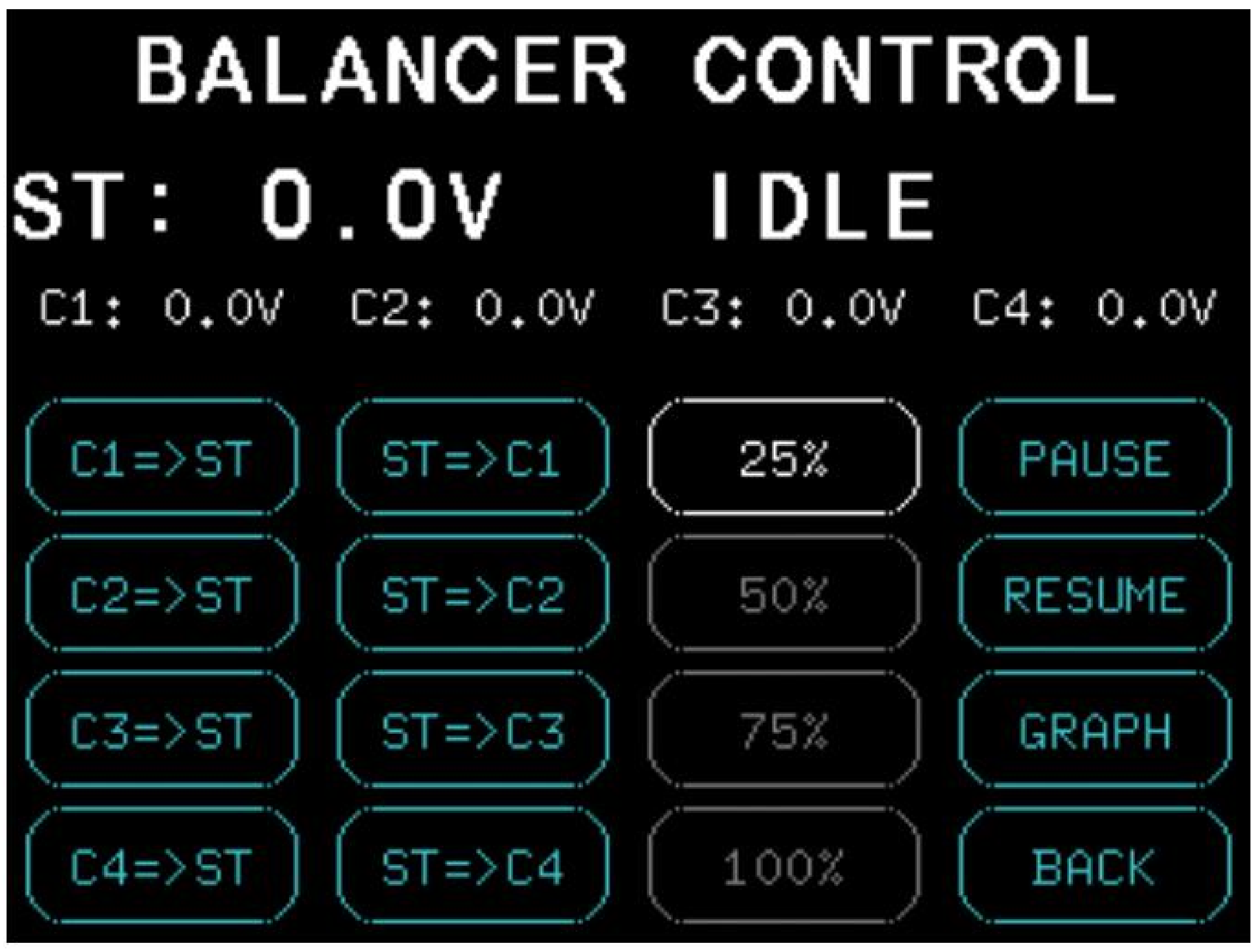


Screen 4: pressing the SWITCHES button on the EDIT TRIGGER page allows the SWITCH OUTPUTs to be set. You can get an idea of the unit's operation from the example configuration we have provided.

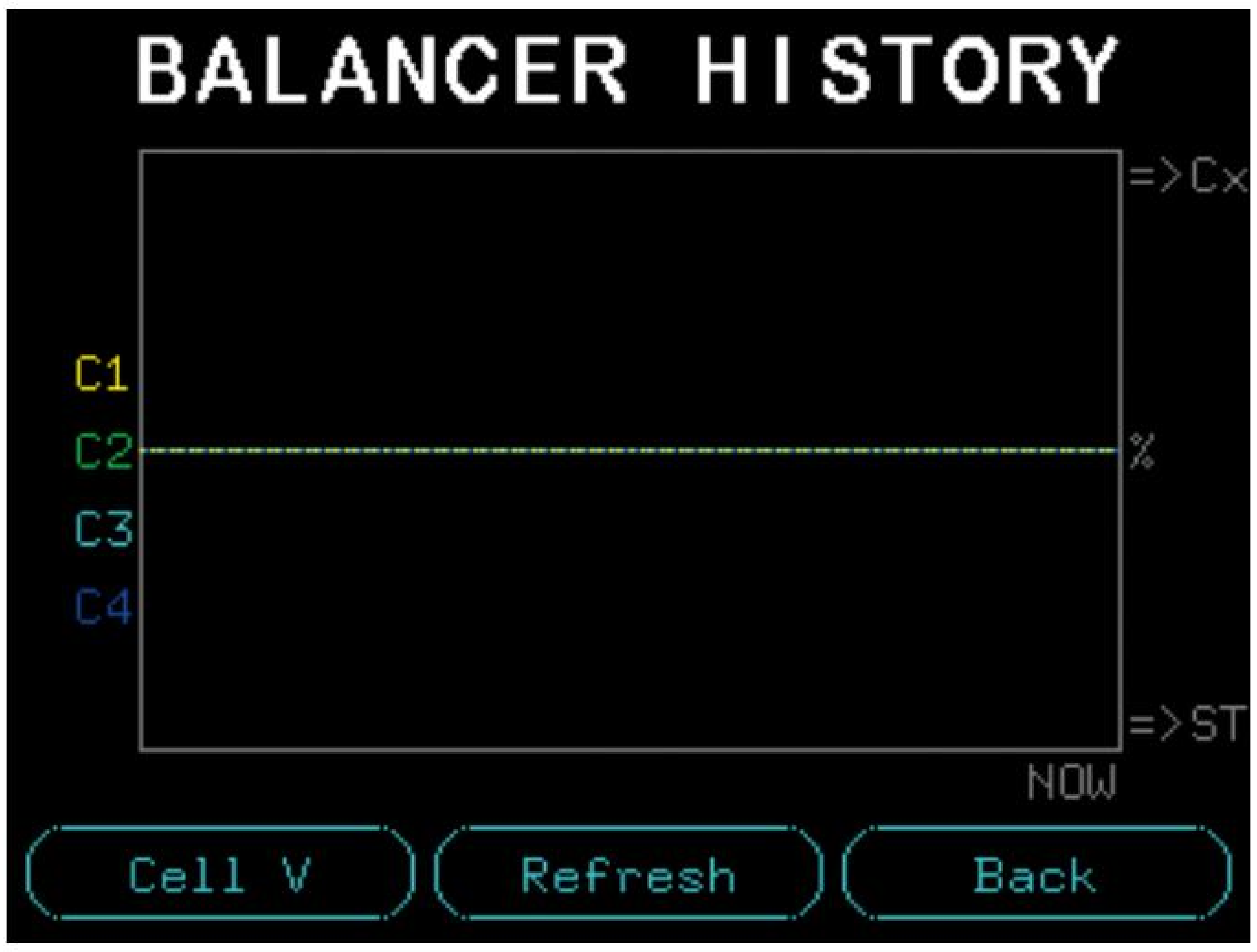


Screen 5: the RESET TRIGGERs are set similarly. All the changes made to these (and other trigger-related) settings are saved on exit from the SOFT SWITCH page.

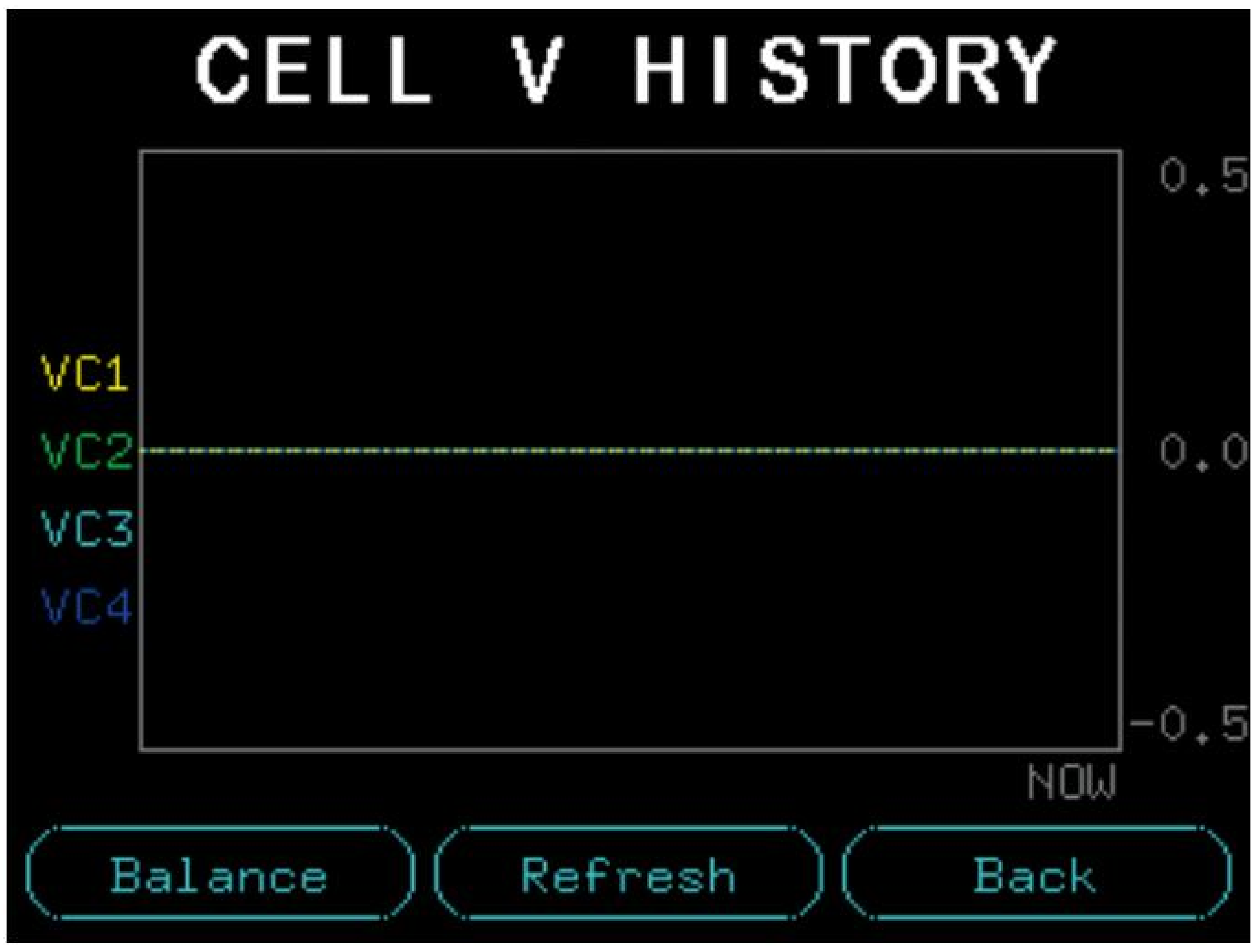
Each trip event can also reset any other trigger, allowing alternate action as two triggers track a variable between the two hysteresis points, as shown by the default settings for TR0 and TR1.



Screen 6: the BALANCER CONTROL page is accessed from the MAIN screen, and shows the current cell voltages and Balancer operating mode. Buttons are provided to issue control commands to the Balancer, assuming it is connected and communicating.



Screen 7: the BALANCER HISTORY page shows the recent operation of the Balancer, including which cells are being balanced and in which direction.



Screen 8: similarly, the CELL V HISTORY page shows the relative cell voltages (to Cell 1). The button at bottom left allows easy toggling between these last two pages.

This is only one way it can be used. Each trigger could be set to require a manual reset or could even reset multiple triggers.

The page shown in Screen 2 lets you manually trip and reset each trigger for testing. Each press toggles between the tripped and reset states.

Pressing the button (such as TR0 for Trigger 0) takes you to Screen 3, which has more settings. The TRIP and RESET buttons work as you would expect. The various buttons labelled V and I allow the threshold variable and condition to be set.

CLEAR removes any threshold, meaning the trigger will not activate automatically. The THRESHOLD+ button sets a positive value, while the THRESHOLD– button is used to set a negative value. This is useful for current thresholds; the *Battery Manager* cannot measure negative voltages.

Finally, the SWITCHES and RESETS buttons allow setting of the actions that result from each trigger.

Screen 4 shows the switch controls; these correspond to the P0-P7 outputs on the I/O Expander, while Screen 5 shows the reset controls, which correspond to the triggers.

All parameters are saved to Flash memory when you press BACK from the Trigger overview page seen in Screen 2. This provides a good compromise between usability and Flash wear.

Balancer menu

From the Main page, pressing the Balancer button goes to the BALANCER CONTROL page, as seen in Screen 6. The two columns of buttons at left will send commands to the *Battery Balancer* to move charge between specific cells and the entire stack.

The rate at which this happens is set by the third column, with options of 25%, 50%, 75% and 100%; the currently selected value is highlighted.

Similarly, the PAUSE and RESUME buttons send commands to the *Battery Balancer* to pause or resume balancing.

The data displayed at the top of the screen is taken from the Balancer in real time.

The GRAPH button goes to the page shown in SCREEN7, which shows the relative flow in and out of each cell. Around 100 data points are stored, and these are updated in time with the logging software's 10-second cycle.

Thus, around 15 minutes of balancing data is available. It is only stored in RAM, so it is erased if power loss occurs. The screen does not automatically refresh; you need to press the Refresh button.

Pressing the 'Cell V' button changes the graph to display the individual cell voltages measured by the Balancer. The chart is centred on the current bottom cell voltage, as this will always be present. The graph spans 1V from top to bottom, allowing cell voltage variations to be easily seen.

Battery Manager

Thus we have updated the *Battery Multi Logger* to the *Battery Manager*. We expect many people will have different requirements regarding what they will control and how they will connect things to the *Battery Manager*.

Indeed, we expect many people will add the *Battery Manager* to an existing battery installation, perhaps in a car, caravan or boat. And it becomes a relatively simple addition to such a system.

In fact, there are so many features in the improved *Battery Manager* that readers may not even wish to add all of them. But this is easy, as it is entirely modular in construction. We wouldn't be surprised if some people use the I/O Expander or Switch Module in unrelated projects.

Some people may not need the *Battery Balancer* add-on, especially those with 12V batteries that don't require balancing.

Both the I/O Expander and the Switch Module will work fine with 3.3V and 5V logic levels, so they could be used on their own (or together) with other microcontrollers such as the Arduino or Micromite.

Reproduced by arrangement with
SILICON CHIP magazine 2022.
www.siliconchip.com.au

Teach-In 8 CD-ROM Exploring the Arduino

This CD-ROM version of the exciting and popular *Teach-In 8* series has been designed for electronics enthusiasts who want to get to grips with the inexpensive, immensely popular Arduino microcontroller, as well as coding enthusiasts who want to explore hardware and interfacing. *Teach-In 8* provides a one-stop source of ideas and practical information.

The Arduino offers a remarkably effective platform for developing a huge variety of projects; from operating a set of Christmas tree lights to remotely controlling a robotic vehicle wirelessly or via the Internet. *Teach-In 8* is based around a series of practical projects with plenty of information for customisation. The projects can be combined together in many different ways in order to build more complex systems that can be used to solve a wide variety of home automation and environmental monitoring problems. The series includes topics such as RF technology, wireless networking and remote web access.

FREE CD-ROM
ELECTRONICS TEACH-IN 8
£8.99
INTRODUCING THE ARDUINO
Teach-In 8
PLUS...

PRICE £8.99

Includes P&P to UK if ordered direct from us

PLUS: PICs and the PICkit 3 – A beginners guide

The CD-ROM also includes a bonus – an extra 12-part series based around the popular PIC microcontroller, explaining how to build PIC-based systems.

SOFTWARE

The CD-ROM contains the software for both the *Teach-In 8* and PICkit 3 series.

ORDER YOUR COPY TODAY!
JUST CALL 01202 880299 OR VISIT www.epemag.com

ESR Electronic Components Ltd

All of our stock is RoHS compliant and CE approved. Visit our well stocked shop for all of your requirements or order on-line.

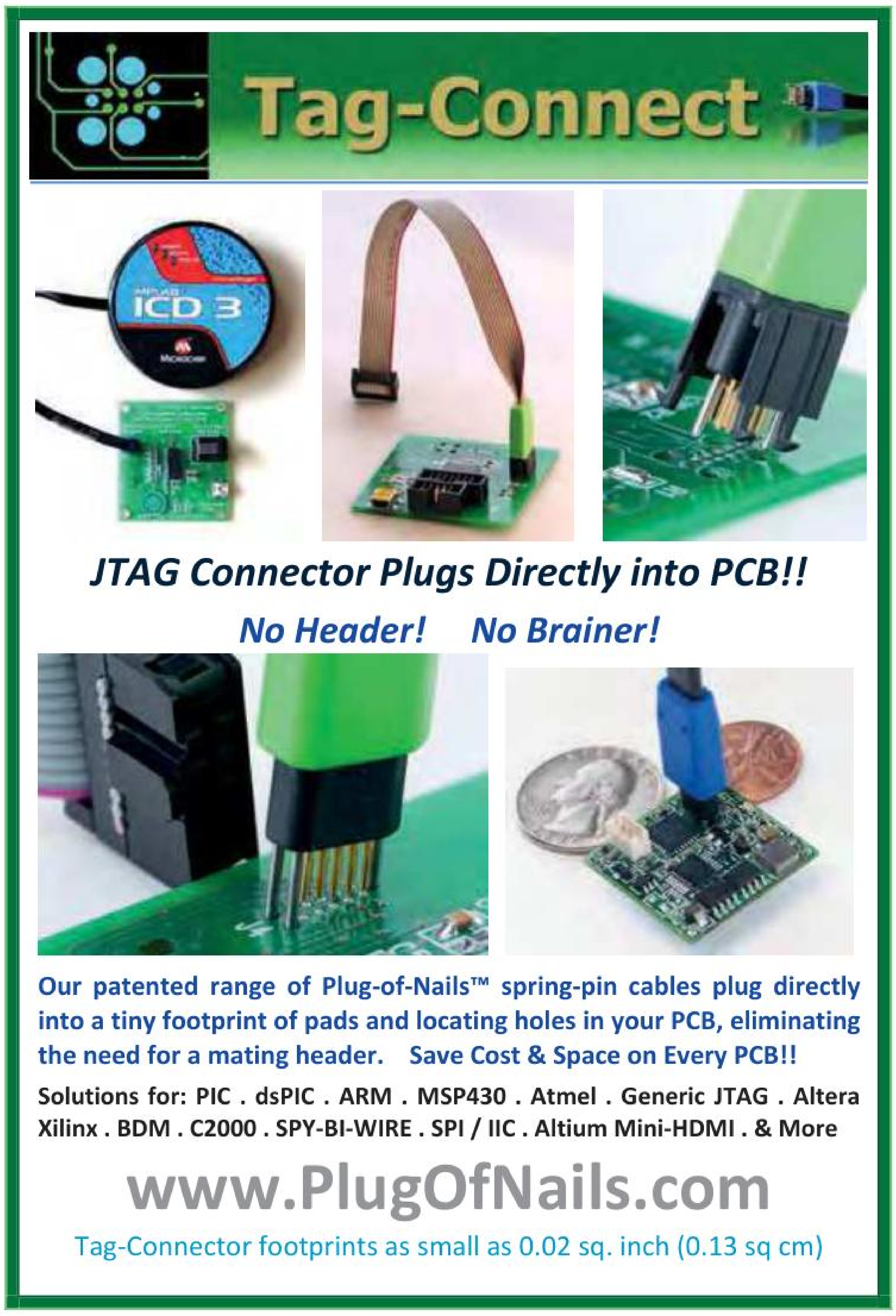
We can help and advise with your enquiry, from design to construction.

3D Printing • Cable • CCTV • Connectors • Components • Enclosures • Fans • Fuses • Hardware • Lamps • LED's • Leads • Loudspeakers • Panel Meters • PCB Production • Power Supplies • Relays • Resistors • Semiconductors • Soldering Irons • Switches • Test Equipment • Transformers and so much more…

Monday to Friday 08:30 - 17.00, Saturday 08:30 - 15:30

Station Road
Cullercoats
North Shields
Tyne & Wear
NE30 4PQ

Tel: 0191 2514363 sales@esr.co.uk www.esr.co.uk



JTAG Connector Plugs Directly into PCB!!

No Header! No Brainer!

Our patented range of Plug-of-Nails™ spring-pin cables plug directly into a tiny footprint of pads and locating holes in your PCB, eliminating the need for a mating header. Save Cost & Space on Every PCB!!

Solutions for: PIC . dsPIC . ARM . MSP430 . Atmel . Generic JTAG . Altera Xilinx . BDM . C2000 . SPY-BI-WIRE . SPI / IIC . Altium Mini-HDMI . & More

www.PlugOfNails.com

Tag-Connector footprints as small as 0.02 sq. inch (0.13 sq cm)

Simple MIDI Music Keyboard

MIDI KEY STRIP
23101213 RevA
©2021
SILICON CHIP
S1
S2
S3
S4
S5
S6
S7
S8

by Tim Blythman

This MIDI Keyboard is a follow-up to our 64-key MIDI Matrix. It is just as flexible and offers a way to easily make music, although it can be repurposed for many other uses.

While MIDI Matrix panels are popular for being a compact way of controlling and interfacing to MIDI equipment, a linear keyboard arrangement like a piano is more ‘standard’ and, for many people, quite intuitive.

This is a modular add-on to the MIDI hardware we introduced in April and May 2022. Like the *MIDI Matrix*, it doesn’t have to be used strictly for MIDI or musical purposes.

The *MIDI Matrix* was designed to be used with an Arduino Leonardo board, since the Leonardo can easily provide a native USB MIDI interface through the versatile Arduino MIDI libraries.

We also demonstrated some program sketches that can run on the Leonardo to give various features, and showed some ways to interface with software on both a PC and an Android smartphone.

At the same time, we presented an Arduino shield that lets you interface the hardware to a great range of MIDI equipment using standard DIN connectors.

This *Keyboard* is intended to replace the *Matrix* as part of a larger construction, as presented in the earlier parts of this series. Refer to those articles, particularly the first part, to understand how the *Matrix* (and now *Keyboard*) can be used. At a minimum, you need an Arduino Leonardo board and some jumper wires to turn the *Keyboard* presented here into a minimal MIDI Encoder.

The Matrix

The original Matrix is basically just an array of pushbuttons that the Leonardo can scan to receive user input. In our MIDI software, each keypress is converted to a musical note.

Each row or column of the *Matrix* is wired to a digital pin on the Leonardo. By using the time-honoured technique of scanning each row in turn, individual button presses can be detected.

In our version of the software, the rows are connected to pins configured as inputs with weak pull-ups. Initially, all column pins are set to a high impedance input mode too.

Each column is configured as an output in turn, and driven low. If any button connected to that column is pressed, its corresponding row pin is pulled down through the switch contacts. By scanning the columns in turn, we can detect individual button presses.

While this system is simple, it cannot detect multiple simultaneous keypresses; for this, each switch needs to be fitted with a diode to prevent ambiguous closures propagating through the *Matrix*. Our *Matrix* omits these diodes in favour of simplicity and compactness, and this linear *Keyboard* is the same in that respect.

The new Keyboard

We considered a linear keyboard for our original design but could not work out a way of making it both compact and functional.

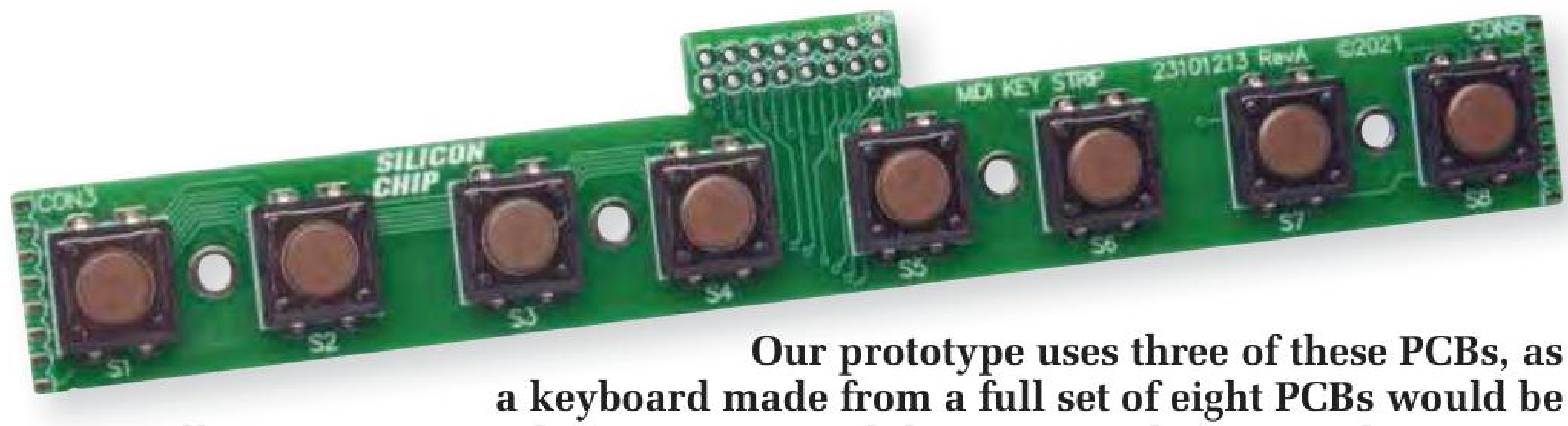
We have now formulated a modular design, so a useful *Keyboard* can be built that is still compact, or it can be expanded to 64 keys, resulting in a device that’s over a metre long! But it still only needs 16 wires to connect it to the Arduino.

The basic unit of the *Keyboard* is a single PCB with eight keys. Each key is wired to the same row contact as the others and also to one of the eight column contacts. A single *Keyboard* module is identical to one row of the *Matrix*.

Fig.1 shows the circuit. CON1 is wired to the columns, with each terminal on CON1 wired to one side of each of the tactile switches, S1-S8. Position 1 of CON2 is connected to the other side of switches S1-S8.

At each end of the *Keyboard* module PCB are connectors CON3-CON6, which can be used to daisy-chain subsequent PCBs to expand the *Keyboard*. These are eight-way surface-mount pads, spaced 2.54mm apart.

CON3 and CON5 (on the top side of the PCB) are wired in the same order as, and in parallel with CON1. Thus, the column signals can pass between the PCBs by joining their adjacent CON3 and CON5. These are wired as a parallel bus.



Our prototype uses three of these PCBs, as a keyboard made from a full set of eight PCBs would be well over a metre wide! We’ve retained the CON1 and CON2 pads on some of the boards to demonstrate and test the different options. In practice, only one set is needed; note that connecting to CON3 and CON4 is equivalent.

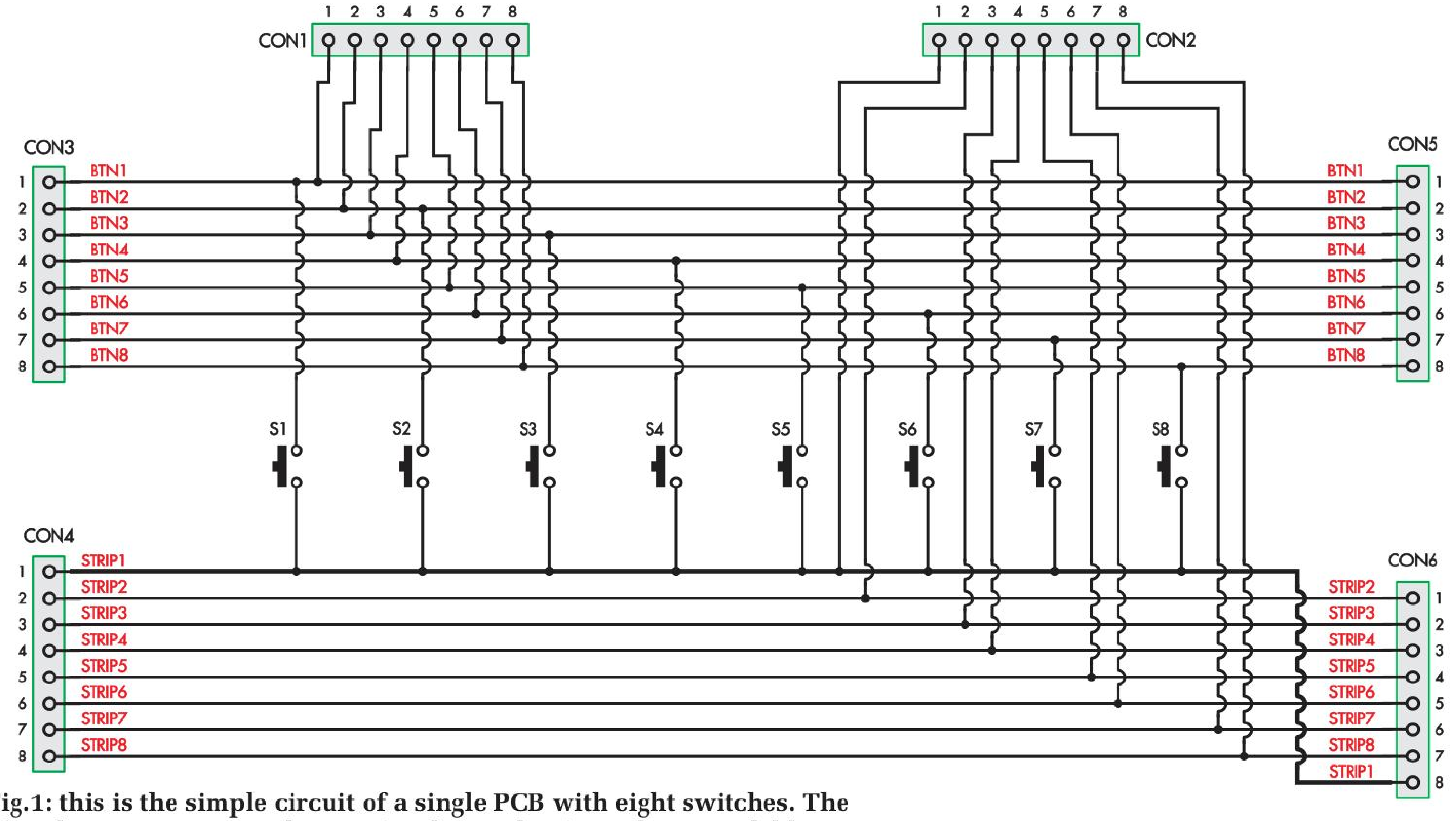

CON1
1 2 3 4 5 6 7 8
CON2
1 2 3 4 5 6 7 8
CON3
BTN1
BTN2
BTN3
BTN4
BTN5
BTN6
BTN7
BTN8
CON5
S1
S2
S3
S4
S5
S6
S7
S8
CON4
STRIP1
STRIP2
STRIP3
STRIP4
STRIP5
STRIP6
STRIP7
STRIP8
CON6
STRIP2
STRIP3
STRIP4
STRIP5
STRIP6
STRIP7
STRIP8
STRIP1

Fig.1: this is the simple circuit of a single PCB with eight switches. The offset between CON4 and CON6 is what makes it easily expandable up to eight PCBs and 64 buttons.

Similarly, on the back of the PCB, CON4 on one PCB connects with CON6 on the next. CON4 is wired the same as CON2, but the clever part is how we have wired CON6. Pin 1 of CON6 is wired to pin 2 on CON4, and so forth, all offset by one position.

Say we wired up an array of eight of these modules, numbering them 1-8 from left to right, with CON3 and CON4 wired to CON5 and CON6 respectively. Connecting to CON1 and CON2 on the first module, we would have the equivalent of a full 8x8 *Matrix* only with the keys in a single row.

Fig.2 shows how the 'rows' are mapped back to CON2 on the first PCB. CON1, CON3 and CON5 are all simply wired in parallel and are not modified by this system.

Other configurations

If you look closely at the PCB, you can see that the little tab where CON1 and CON2 jut out is scored for removal. This lets you remove these tabs on all but one module.

In fact, since CON3 and CON4 are wired identically to CON1 and CON2, you can even remove the tab from all boards and simply take the matrix connections from CON3 and CON4 of the leftmost board instead.

If you don't mind remapping the pins in software (or changing how they are wired back to the Leonardo board), the CON1 and CON2 connections do not have to be made on the first board. You could even take these connections from the middle of the array.

We've designed the PCB to use large 12mm tactile switches, as these have a much nicer feel with a large finger surface. You might find that some smaller switches can be made to fit by bending their leads, although we haven't tried that.

Since there is less space for routing on this PCB than the *Matrix*, it lacks the option to fit illuminated switches that the *Matrix* had.

Hardware

Like the *Matrix*, the *Keyboard* we are presenting has quite a basic design, so that you can customise it to your requirements. The switches are placed on 20mm centres, with four M3 mounting holes provided on each PCB.

Nominally, the mounting holes will be on 40mm centres, although this depends on the accurate assembly of adjacent boards. The PCBs are 20mm

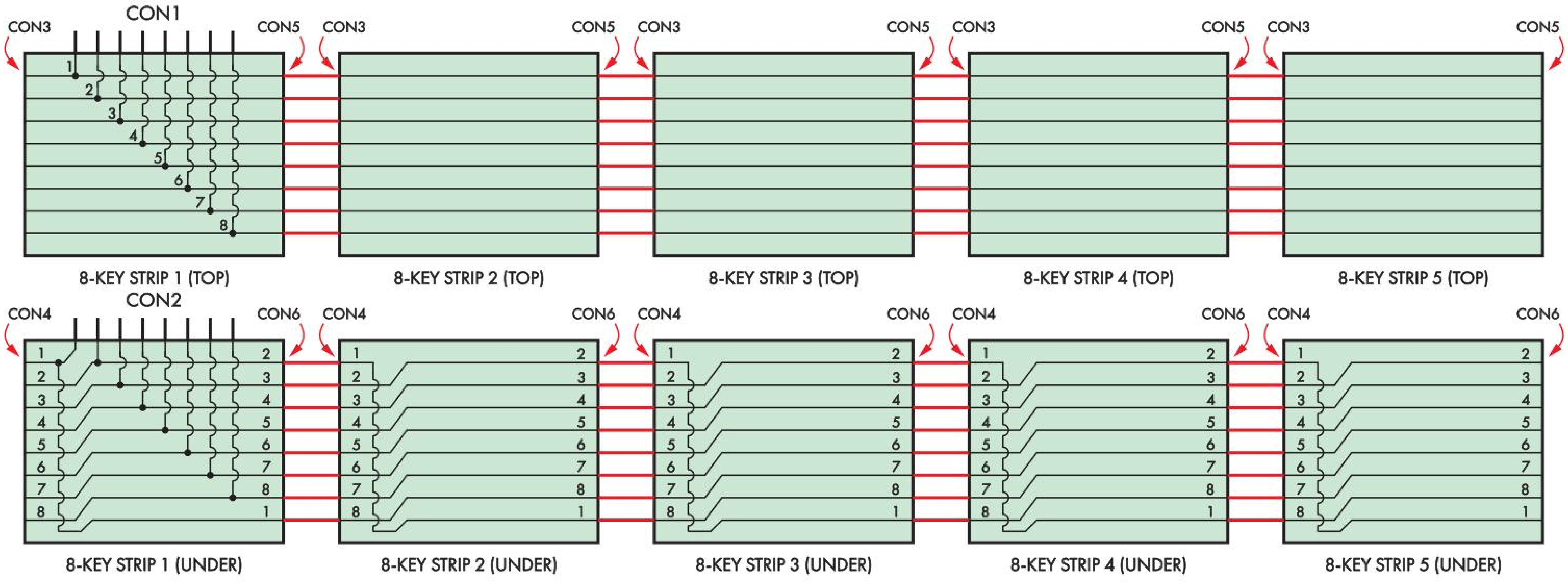

CON1
CON3
CON5
1
2
3
4
5
6
7
8
8-KEY STRIP 1 (TOP)
8-KEY STRIP 2 (TOP)
8-KEY STRIP 3 (TOP)
8-KEY STRIP 4 (TOP)
8-KEY STRIP 5 (TOP)
CON2
CON4
CON6
1 2 3 4 5 6 7 8
2 3 4 5 6 7 8 1
8-KEY STRIP 1 (UNDER)
8-KEY STRIP 2 (UNDER)
8-KEY STRIP 3 (UNDER)
8-KEY STRIP 4 (UNDER)
8-KEY STRIP 5 (UNDER)

Fig.2: this shows how multiple 8-button Keyboard PCBs are joined so that the Arduino can tell which key has been pressed. Each PCB along the chain offsets where the connection is ultimately made at CON2, allowing for up to 64 keys to be sensed.

PDF NEWSPAPERS and MAGAZINES: WWW.XSAVA.XYZ & WWW.SPICYMAGS.XYZ

high, not counting the tab for CON1 and CON2; 28mm with the tab in place.

We strongly recommend mounting the *Keyboard* to a sound backing so that the PCBs do not flex when the keys are pressed. The connections for CON3-CON6 will not provide much mechanical strength as they are effectively surface-mounting pads, and are only bonded to the PCB superficially.

What about the black keys?

You might be thinking that pianos actually have two rows of keys, white and black, and you would be right. Also, there are seven white keys per octave, not eight. We have kept this as a linear array of eight keys to make it simple and applicable to a wide range of applications.

We plan to produce a 12-key PCB at a later date which has the keys staggered and grouped like a piano. In the meantime, if you're keen to use this board like a proper piano, you could build it in two rows, with the top row offset horizontally 6mm from the bottom row and with gaps in the keys at the top to give the proper configuration. Both rows could be wired up in series (assuming they contain no more than 8 PCBs total).

The software could be modified relatively easily to remap the two rows of keys into the correct sequence so that it can act as a keyboard piano. The restriction of only one keypress being detected at a time would remain, though. Our planned future piano keyboard PCB would remove that restriction.

Construction

Most people will want to build a *Keyboard* with multiple PCBs laid out as a continuous strip, so we will describe what is needed to achieve this. The Keyboard is built on a PCB coded 23101213, which measures 158 x 28mm and is available from the *PE PCB Service*. Use the PCB overlay diagram, Fig.3, as a guide to fitting the components.

Plan and lay out the modules before commencing construction. To keep things compact, the connections between the boards are a little tight, and it will be easier to join them before fitting other components.

If you want a different layout, just about any method of wiring CON3 and CON5, and CON4 and CON6 respectively will work. You might even like to use header sockets on one and header pins on the other to allow the units to be unplugged, although this will not achieve a tight spacing.

To start the PCB assembly, snap off any CON1/CON2 header tabs that are not needed. Do this by scoring along the line with a sharp knife to cut the copper traces, then carefully flex the PCB with pliers to make a clean break.

You might like to clean up the rough edge. As well as our usual warnings about avoiding the inhalation of PCB dust (eg, by working outside and wearing a mask), take care not to file away the traces which run close to the edge of the PCB, especially at the back.

Each PCB is 158mm long, meaning that there is 2mm of spare space for a joiner if the key spacing is to be kept even. We used cut-down double-row pin headers. The plastic spacers are very close to 2mm deep, giving the necessary spacing.

Start by cutting down the headers to be used for joiners. This is fiddly but necessary, as there is no more than 8mm between adjacent switch bodies on neighbouring PCBs, and typical pin headers are around 11mm tall.

You can halve the number of cuts by shifting the pins in the plastic. Place the PCB on a hard flat surface and rest the 2x8 pin header in the CON1/CON2 holes. Push the plastic down firmly with a flat edge that fits between the pins. A steel ruler is ideal for this.

This will move the pins such that only 1.6mm (the PCB thickness) of each pin is proud. Now reverse the 2x8 pin header, and use the depth of the PCB as a jig to cut 1.6mm from the other side of the pins.

The pin stubs may fly off at speed, so wear safety goggles and aim the header while cutting so that they fly away from you. See the photos opposite that show what the header should look like after being trimmed and then attached to the PCBs.

Soldering these headers is a little tricky as they are not a snug fit. Treat them like a surface-mounted part, applying flux paste to the pads before soldering. We recommend securing the parts during soldering with high-temperature tape (eg, Kapton) so they don't move around.

Tack the ends in place and check that the pins do not foul the tactile switch footprints. You might even like to test-fit the switches to confirm clearances.

Solder the remaining pins – be generous with the flux, it helps the solder form clean beads that sit where they need to. Flip the board and complete the headers on the back of the PCB.

Remove any excess flux using a flux cleaner, and test the exposed CON3-CON6 pads for continuity between the ends of the strip. As you can see from Fig.2, CON3 is wired straight through to CON5. But CON4 will be offset relative to CON6 (unless you have the full complement of eight PCBs), so check that each pad on CON4 is connected to one and only one pad on CON6.

It's best to do this now, as it can be quite fiddly to rework these connections with the tactile switches in place.

Fit the switches next. They should snap neatly into place; just check that they are sitting flush before soldering.

Finally, solder headers for CON1 and CON2. We used female headers to match the cables we had made up for the *Matrix*, but you can use whatever works for your arrangement, even soldering wires directly to the PCB.

Hooking it up

We tested our unit with the MIDI_ENCODER sketch. If you haven't done so already, we recommend reading the earlier parts of this series of articles, as they describe the software in more detail.

Since the *Keyboard* is effectively equivalent to a *Matrix* fitted with non-illuminated switches, you can transfer many of the ideas relating to the *Matrix* to the *Keyboard*.

As with the *Matrix*, wire CON1 of the *Keyboard* to CON2 of the MIDI shield (or corresponding Leonardo pins) and CON2 of the *Keyboard* to CON1 on the MIDI shield, connecting pin 1 to pin 1.

Check that all buttons work as expected using the key notifications that appear on the Arduino Serial Monitor. If you find that some keys on a PCB don't work (but not all), check the column connections for continuity; these are CON3 to CON5 on the front of the PCB.

If none of the keys on a PCB work, then it may be a problem with the CON4 to CON6 row connections on the back of the PCB.

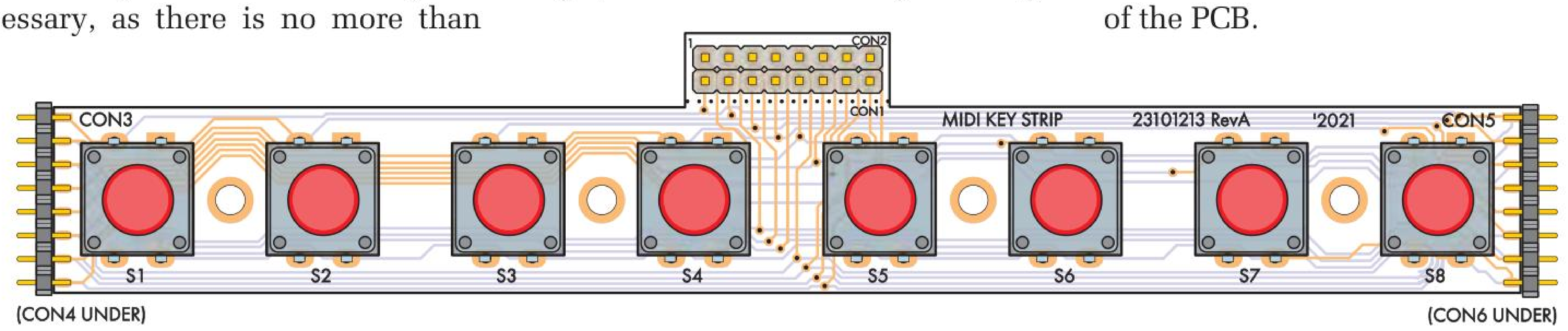
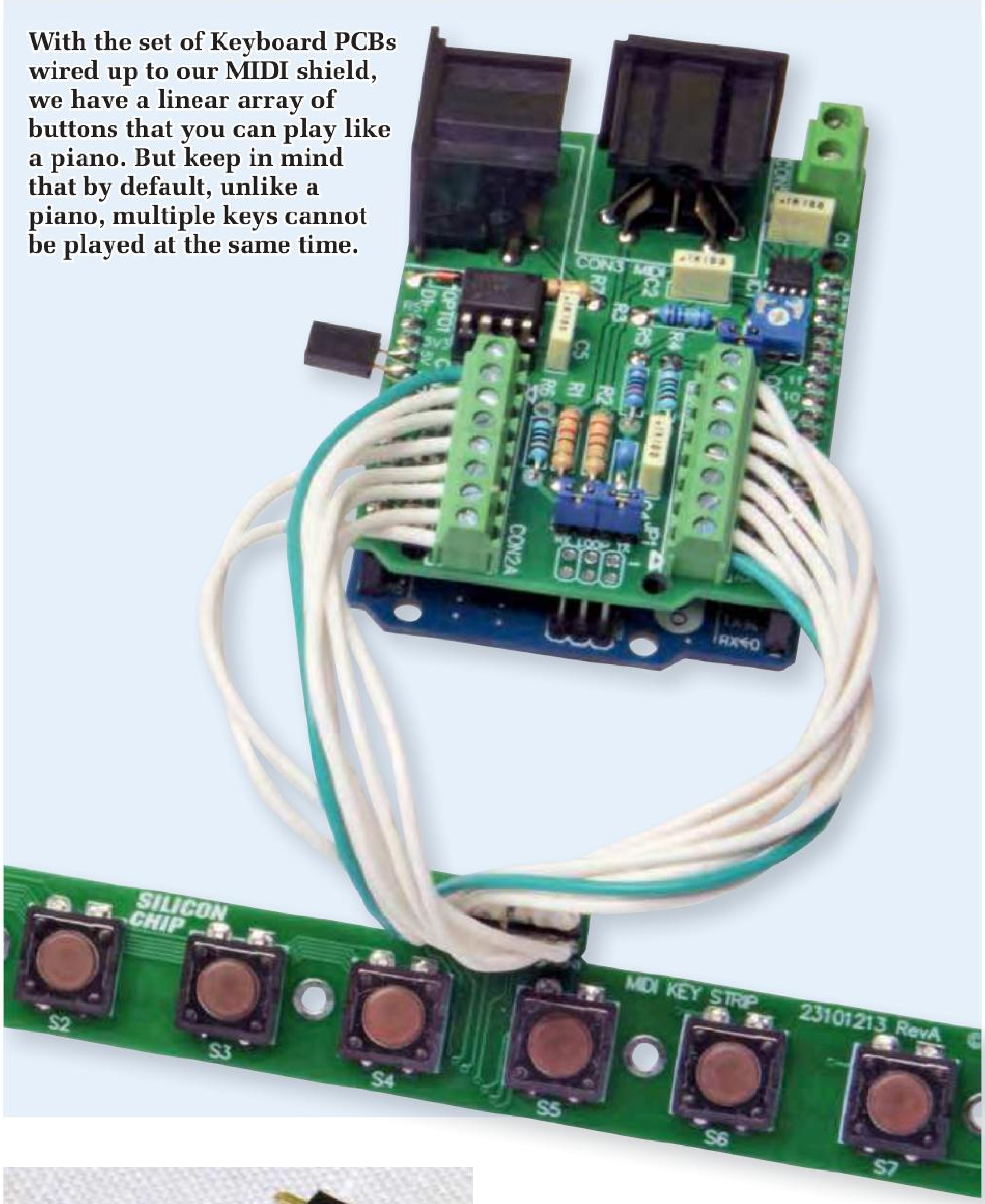


Fig.3: there's not much to get wrong during assembly, although we recommend fitting the PCB joiners first, as the tactile switches will make access difficult when soldering them. The buttons should snap into place, so soldering them is easy.

Parts List – Full 64-key Keyboard

8 Keyboard Modules
7 2x8 male pin headers, trimmed in height (CON3-CON6)
1 2x8 pin header (male or female to suit Leonardo connections, CON1 and CON2)
mounting hardware to suit usage (M3 tapped spacers, screws etc)

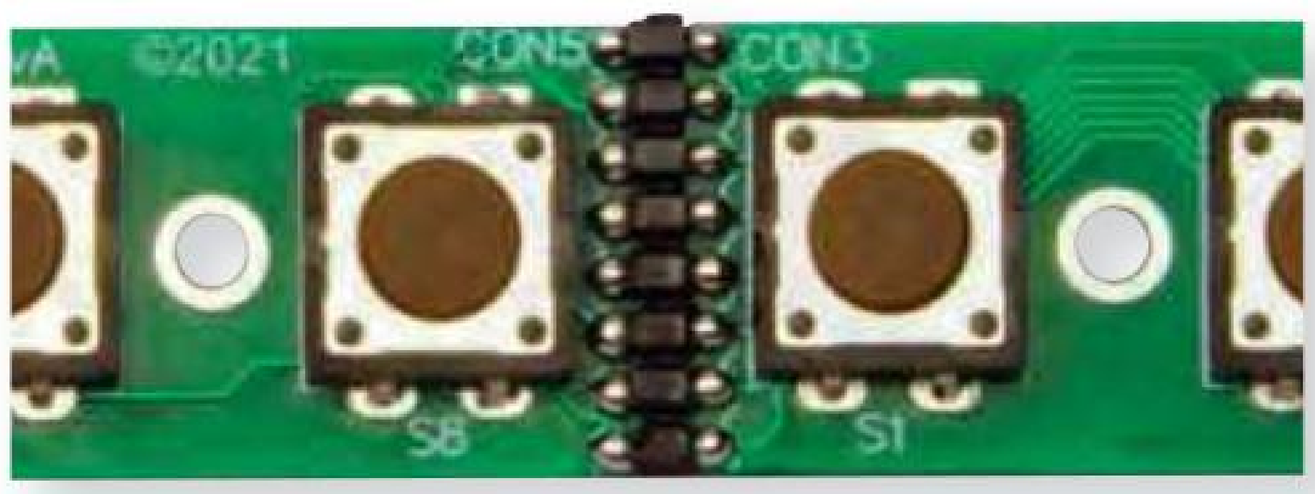
Keyboard Module
1 double-sided Keyboard PCB coded 23101213, 158 x 28mm (available from the PE PCB Service)
8 12mm tactile switches **[eg, Diptronics DTS-21N-V or Jaycar SP0608/SP0609, Altronics S1135 + S1138]**


With the set of Keyboard PCBs wired up to our MIDI shield, we have a linear array of buttons that you can play like a piano. But keep in mind that by default, unlike a piano, multiple keys cannot be played at the same time.

Conclusion

Like the *Matrix*, the *Keyboard* is designed to work with our MIDI hardware and software. But we think that readers will find other uses, especially in cases where many buttons need to be connected to a microcontroller.

Reproduced by arrangement with
SILICON CHIP magazine 2022.
www.siliconchip.com.au



The cut-down header pins (above-left) measure around 7mm tall so that they will fit between the end switches on adjacent PCBs (below-left). The plastic part is 2mm tall, so uniform board spacing is achieved too.

PoLabs
For more information or software download please visit
www.poscope.com/epe

PoKeys
Connect, Control
- USB
- Ethernet
- Web server
- Modbus
- CNC (Mach3/4)
- IO
- PWM
- Encoders
- LCD
- Analog inputs
- Compact PLC

Stepper motor drivers
Drive
- up to 256 microsteps
- 50 V / 6 A
- USB configuration
- Isolated
- up to 32 microsteps
- 30 V / 2.5 A

PoScope Mega1+
PoScope Mega50
Measure
- up to 50MS/s
- resolution up to 12bit
- Lowest power consumption
- Smallest and lightest
- 7 in 1: Oscilloscope, FFT, X/Y, Recorder, Logic Analyzer, Protocol decoder, Signal generator

Nano Pong on your TV

By Tim Blythman

Atari's Pong arcade game is nearly 50 years old and is remarkable for its time, inspiring many of the computer games that followed. Our Nano Pong game is modern and retro at the same time; it replaces the 70-odd discrete logic chips in the original with a single chip that costs about £1! But it still looks and plays much like the original game.

If you've ever seen any of the modern single-chip versions of Pong, you may well have felt they lacked a certain something and were inferior to the original 'discrete' (multi-chip) incarnation. This project is an attempt to change that!

While this is a complete redesign of the circuitry to implement the Pong game, we have tried to be reasonably faithful to the original in terms of its graphical style, and how the game is played.

Another inspiration for this design is the power of new 8-pin PIC microcontrollers. One such chip that we like is the PIC12F1572 microcontroller. It's one of the cheapest 8-pin PICs available, but despite low cost, it has superior features to the PIC12F675 that we have used for many years in out projects.

As one of the smallest, cheapest microcontrollers around, we decided that it would be an interesting challenge to use it to recreate Pong.

Nano Pong

Our version of this classic game is made using not much more than a small microcontroller and some passive components. It's so tiny that we haven't even specified a case for it; it can simply be wrapped up in a length of heatshrink tubing and left hanging behind the TV.

A pair of controllers ('paddles') are built into small enclosures on flying leads, but if you're interested in creating something more akin to the cabinets and consoles that would have existed at the time, you can do that too.

Nano Pong is closely inspired by the original Pong; two players control on-screen bats that vie to keep the ball in play. The winner of the game is the first to win 11 rallies. We say 'inspired' because we haven't attempted to make it identical. No doubt those who played the original game would notice some differences.

But we have tried to emulate the style and gameplay of the older game. In so doing, we hope that those building this project can experience the joy of playing a 50-year-old computer game without the hassle of having to locate and solder a multitude of vintage logic chips.

Like the original, the two player control paddles are potentiometers that translate the player's paddle position to a corresponding on-screen paddle.

We've also added a pushbutton (which isn't in the original) to allow a player to 'serve' the 'ball'. The PIC chip emulates the gameplay mechanics, and generates analogue audio and video signals that can be fed to a PAL television's AV inputs.

Hardware

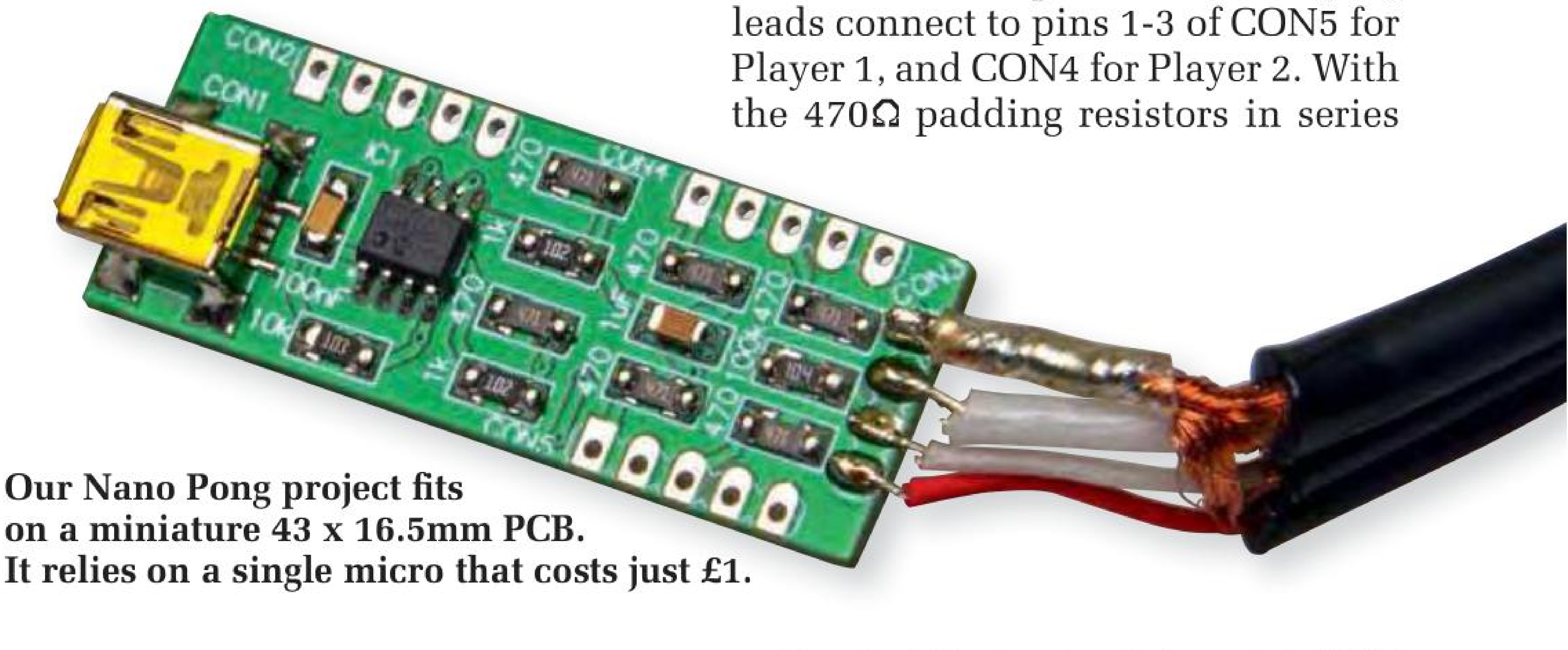
Fig.1 shows the complete schematic – there is not much to it! Potentiometers VR1 and VR2 and pushbuttons S1 and S2 are not located on the PCB, but connected via flying leads.

CON1 is the first modern flourish. A mini-USB socket provides 5V power to the circuit, significantly simpler than the original mains supply. Since many TVs now have a USB socket, the unit can be powered from the TV that it's connected to.

5V power goes to pins 8 (GND) and 1 (V_{DD}) of IC1, a PIC12F1572 microcontroller, bypassed with a 100nF capacitor. IC1's $\overline{\text{MCLR}}$ pin is pulled up to 5V by a 10kΩ resistor, so the PIC will run its internal program from Flash memory as soon as power is applied.

Pins 7 and 6 of IC1 are inputs to the ADC (analogue-to-digital converter) peripheral and communicate the Player 1 and Player 2 control inputs to the PIC.

Each player has a 1kΩ potentiometer padded on both track ends by a 470Ω resistor. The resistors are fitted to the PCB. The potentiometer flying leads connect to pins 1-3 of CON5 for Player 1, and CON4 for Player 2. With the 470Ω padding resistors in series



Our Nano Pong project fits on a miniature 43 x 16.5mm PCB. It relies on a single micro that costs just £1.

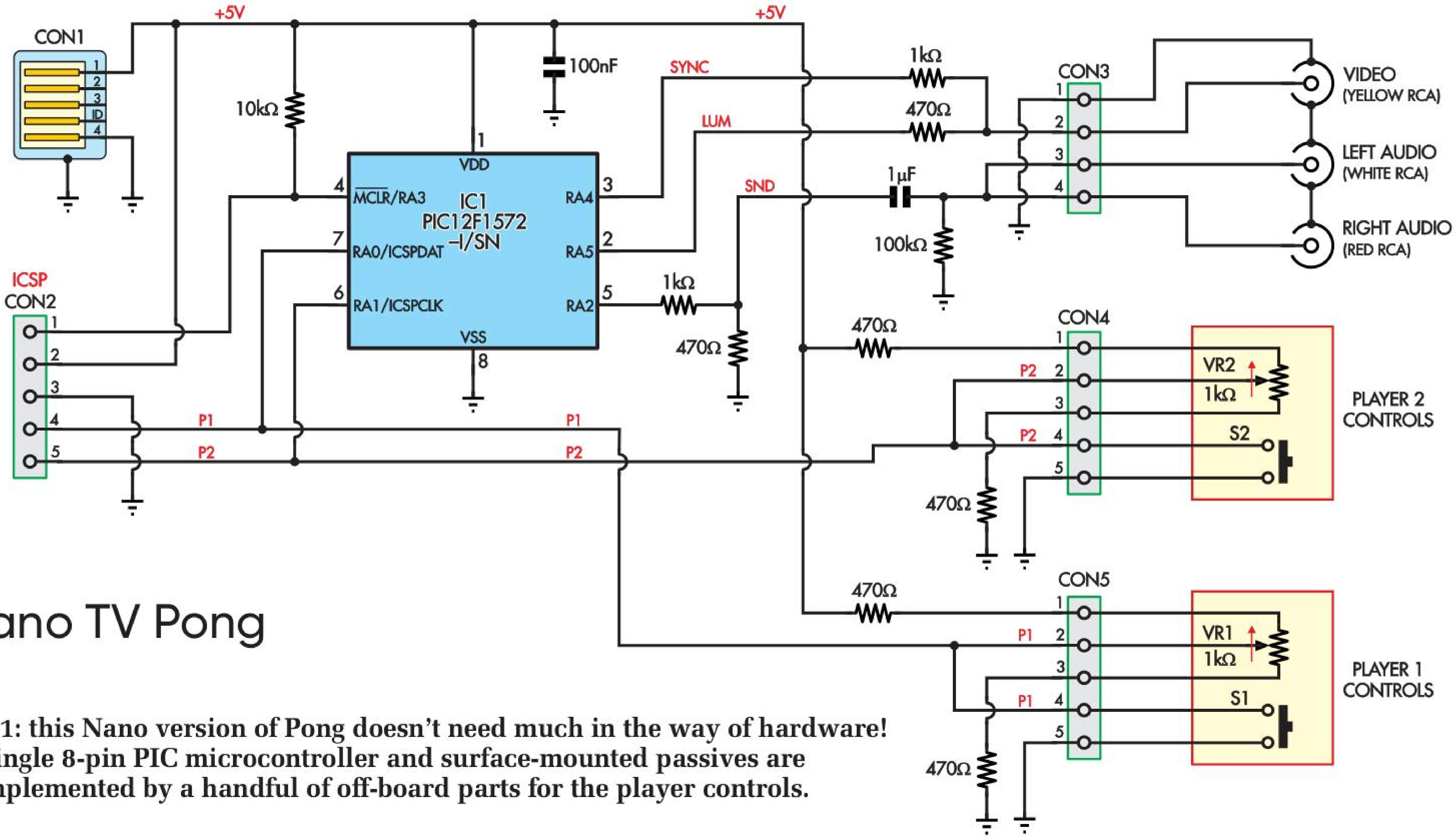


Fig.1: this Nano version of Pong doesn't need much in the way of hardware! A single 8-pin PIC microcontroller and surface-mounted passives are complemented by a handful of off-board parts for the player controls.

with the 5V supply, the player paddle wipers vary between 1.25V and 3.75V depending on the potentiometer rotation.

We've specified standard 24mm potentiometers, but if you think your *Nano Pong* might be subjected to extended periods of vigorous gameplay, you could upgrade them to more robust types.

Although it would not be in line with the original, slide (rather than rotary) potentiometers could also be used to make a more intuitive interface, matching the straight-line motion of the bat on the screen.

Pins 4 and 5 of CON4 and CON5 connect across NO (normally open) momentary pushbutton switches. By pressing the button, the player pulls pin 2 of CON4/CON5 (connected to the potentiometer wiper) to 0V. As the lowest voltage the pot can generate is around 1.25V, the microcontroller can distinguish this as a button press. The top padder resistor limits the worst-case current through the switch.

Pin 5 of IC1 is the pulse-width modulated (PWM) sound output. It feeds a 1kΩ/470Ω divider, reducing the PWM amplitude from 5V peak-to-peak to around 1.6V peak-to-peak or 0.56V RMS. This is AC-coupled by a 1µF capacitor and biased to ground by a 100kΩ resistor before going to the output RCA plugs that connect to the TV.

We've chosen these values to keep the sound signal well below 1V, as the audio (as per the original Pong) is a shrill-sounding square wave.

The video signal is a standard CVBS (composite video baseband signal) in monochrome PAL format. Many of the differences between PAL and NTSC involve colour transmission, so many NTSC TVs should lock onto this signal. The main remaining difference is in the number of lines that are sent per frame. Modern TVs will usually detect and display the correct format.

This signal is formed from digital levels at output pins 2 and 3 of IC1. Pin 2 is designated as luminance (LUM) and pin 3 as synchronisation (SYNC). The TV is assumed to have a 75Ω terminating impedance, so it will see different voltages depending on the pin states.

If both LUM and SYNC are low, then the output is 0V, which corresponds to the so-called 'sync' level. With SYNC high and LUM low, the TV sees around 300mV. This is known as black level, and corresponds to a black raster being displayed. Finally, with both pins high, a level near 1V is seen, which generates a white raster.

Scope 1 shows the voltage generated over time for a typical horizontal scan line, along with several lines of video (including this scan line) above. Note the horizontal sync pulse troughs on either side of the displayed video.

Analogue video signal

So how does a TV translate this signal to a two-dimensional picture? The TV continually scans its raster in left to right horizontal scan lines from top to bottom of the screen, with each scan line taking around 64µs.

A 4-5µs low pulse indicates the start of a new horizontal line. The visible area takes up most, but not all, of the remaining scan line. The actual visible area takes 52µs to transmit, so it is bracketed by periods of black level called the 'back porch' and 'front porch'.

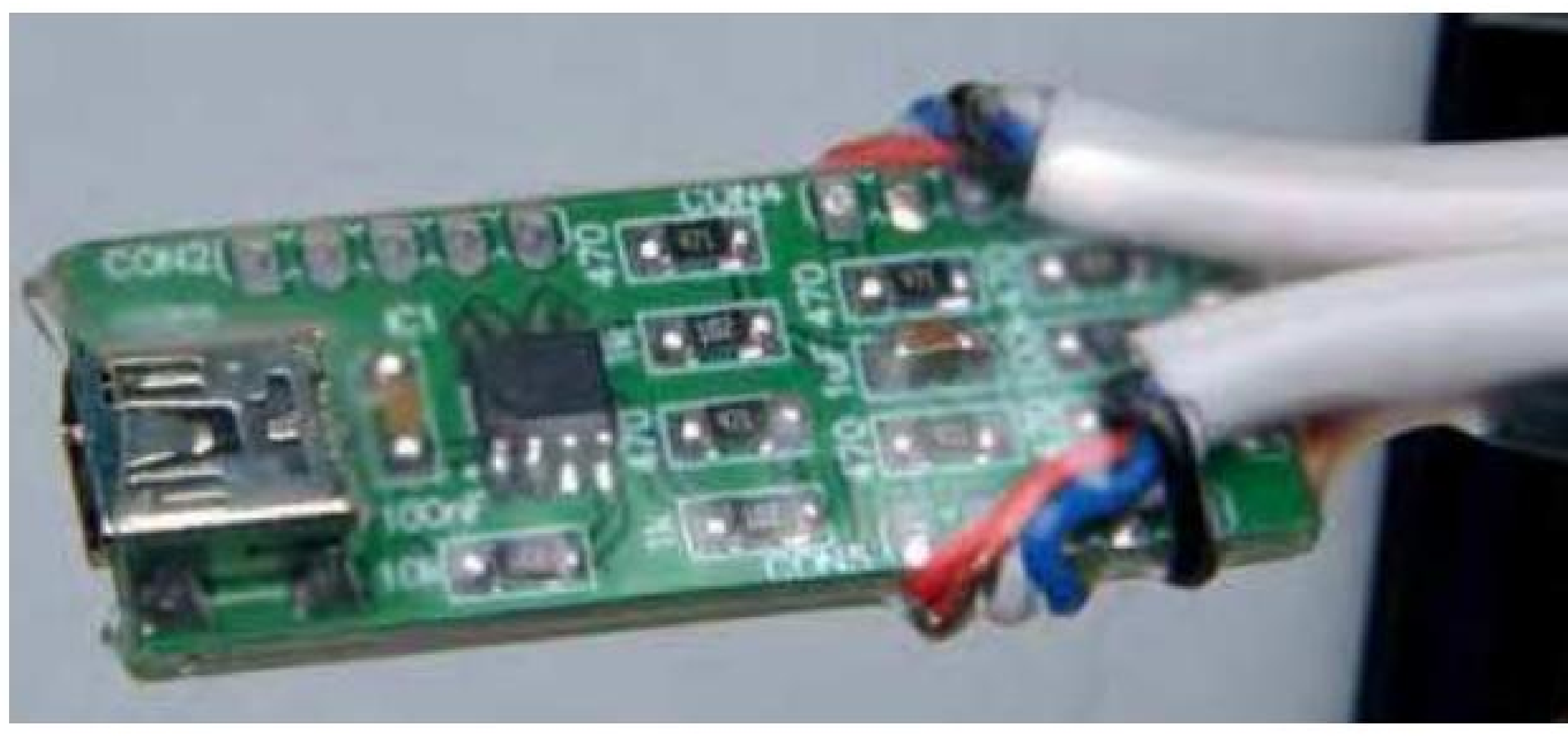
Colour transmissions contain signals in the back porch to help decode the colour picture information; since we are not transmitting such signals, the picture is decoded as monochrome. During the visible video area, the video intensity is determined by the signal voltage, between the black and white levels.

A longer sync signal is used to trigger a vertical retrace. Often, the vertical sync signal is mixed with the horizontal sync signal to create a so-called 'serrated' sync signal that allows horizontal sync to be detected during the vertical retrace. This improves the TV's ability to maintain horizontal hold.

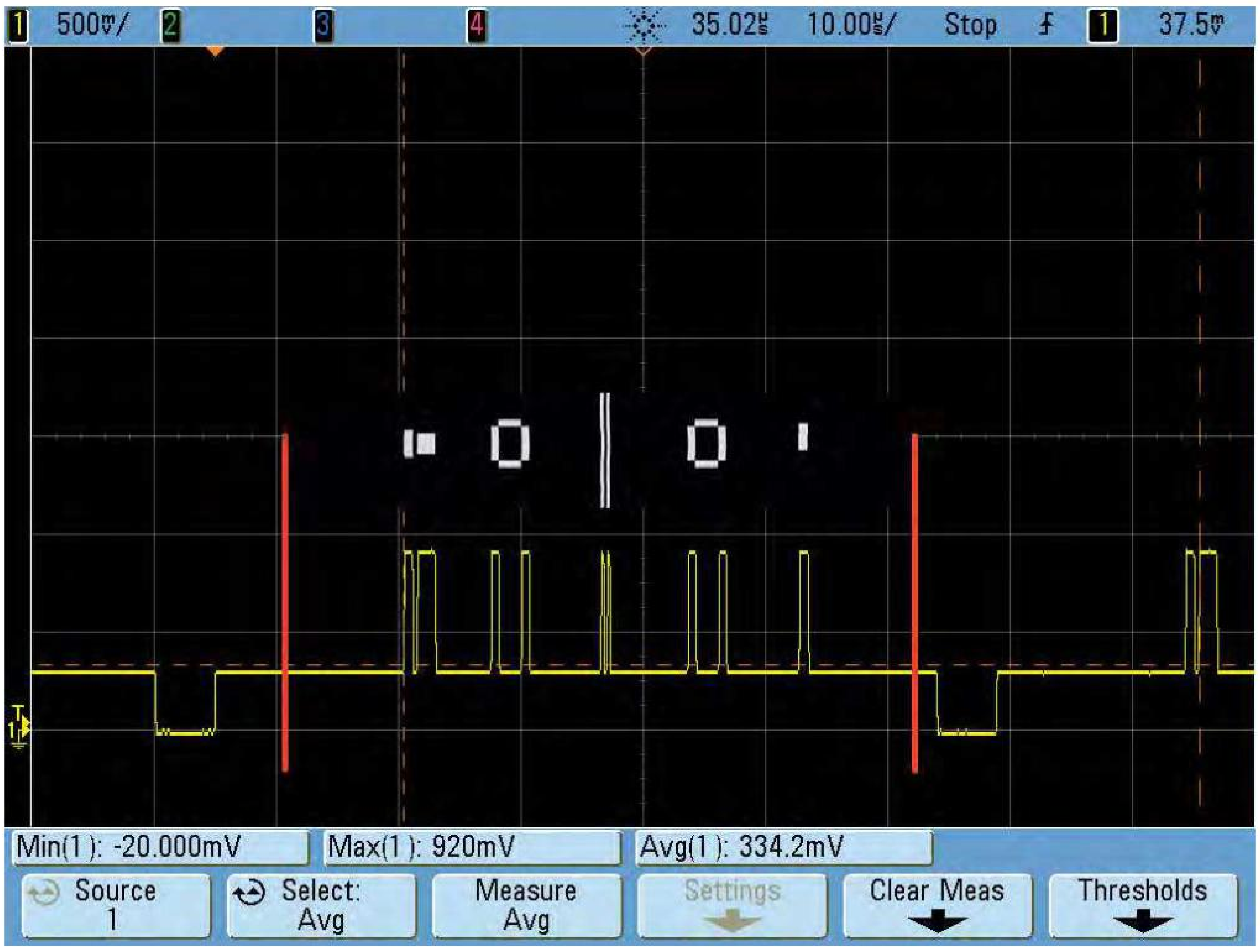
Thus, a single 1V peak-to-peak analogue signal can encode raster intensity and both horizontal and vertical synchronisation to recreate a 2-D TV image.

Software

Understanding the following is not necessary for getting *Nano Pong* to work. Still, it is interesting to compare it with how the original version



Once finished, the PCB and cabling can be covered with heatshrink tubing.



Scope 1: a scope grab of the video signal for a typical scan line, along with a portion of the display around that scan line, so you can see how they correspond. Each line is delimited by the dips in the trace to the low sync level (horizontal sync pulses), while the peaks correspond to a white raster on a black background. The red lines indicate that there is a substantial part of the signal outside of the visible area.

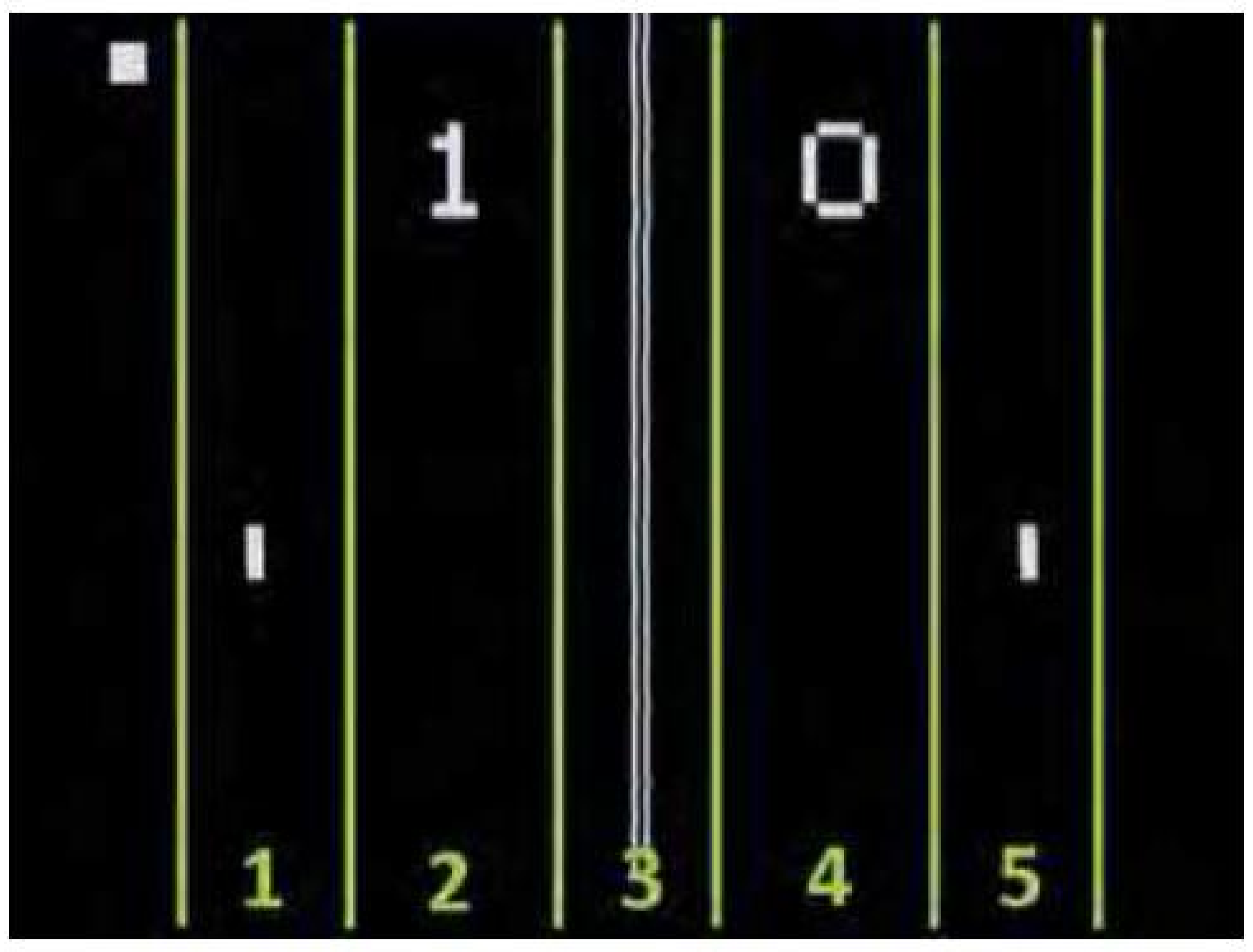


Fig.2: the way that the screen is laid out makes it very easy to generate in a left-to-right fashion. Each horizontal scan line can display the Player 1 bat, Player 1 score, net, Player 2 score and Player 2 bat. The ball is produced separately by the PWM peripheral so that it can appear at any horizontal position.

operates, especially since the original version was purely hardware-based.

Accurately emulating the logic chips in Pong would be a better job for an FPGA than a microcontroller, as the former allows everything to happen independently in parallel, a luxury we do not have.

Our PIC needs to generate a tightly timed signal to maintain a steady picture. Most older PICs require a crystal oscillator to provide an accurate enough clock to display a TV image, which would take up two of our eight pins.

But here, we get accurate timing by running the PIC's internal oscillator at 32MHz (requiring the use of the internal PLL), which gives an instruction clock of 8MHz. At 64µs per line, we can distinguish up to 512 horizontal positions per line. By scaling this to use 256 positions, we can use 8-bit bytes to hold pixel locations. In practice, the actual horizontal play area is around 200 positions.

If you look closely at our images, there is a bit of horizontal jitter, which would not be present with a more precise crystal oscillator. But we don't think it looks out of place in our recreation of 50-year-old technology.

As PAL TV signals have 312 horizontal lines per field, we conveniently set the play area to be 256 lines, which neatly lines up with the visible area on most TVs.

With such tight timing needed, we have fallen back to using assembly language so that we know how long every part of our program will take to execute, ensuring the image quality does not suffer.

The initial setup is written in the C language. It then calls our main assembly language subroutine. The main program is a loop of 312 subroutine calls, each corresponding to a horizontal display line variant. These, in turn, consist of numerous direct pin manipulation commands to set the necessary video output levels interspersed with calls to a delay routine to affect the timing.

This starts with six vertical sync lines to start the field, followed by 28 blank lines. The blank lines are 5µs at sync level (SYNC and LUM low), followed by 59µs at black level (SYNC high and LUM low). The vertical sync line is serrated by delivering 5µs of black and 59µs of sync instead.

After this there are 256 active display lines. The counter LINECOUNT is used to keep track of which line is being displayed, and this is compared with the bat and ball positions, then flags are set to indicate whether the bat or ball should be displayed on the current line. These flags are set during the line's horizontal sync period, so it does not affect the timing of the visible part of the display. The way that these flags are set is somewhat unusual.

To ensure that each line runs for the same amount of time, as needed to maintain a steady picture, we avoid skipping over code we don't want to run (as usually happens if a condition is false), which would change the program timing.

Instead, we use the 'skip on bit test' assembler opcodes (BTFSC and BTFSS), which essentially treat the following opcode as a NOP (no operation) if a test is true. These sequences of commands all take the same time regardless of their outcome, retaining the necessary consistent timing.

For lines where the ball is visible, we use the PWM peripheral to display it. The PWM peripheral on the PIC12F1572 is quite advanced, with phase and offset parameters. The ball's horizontal position is determined by the PWM phase and its width by its duty cycle. This means we don't have to keep track of when to turn the LUM output on and off.

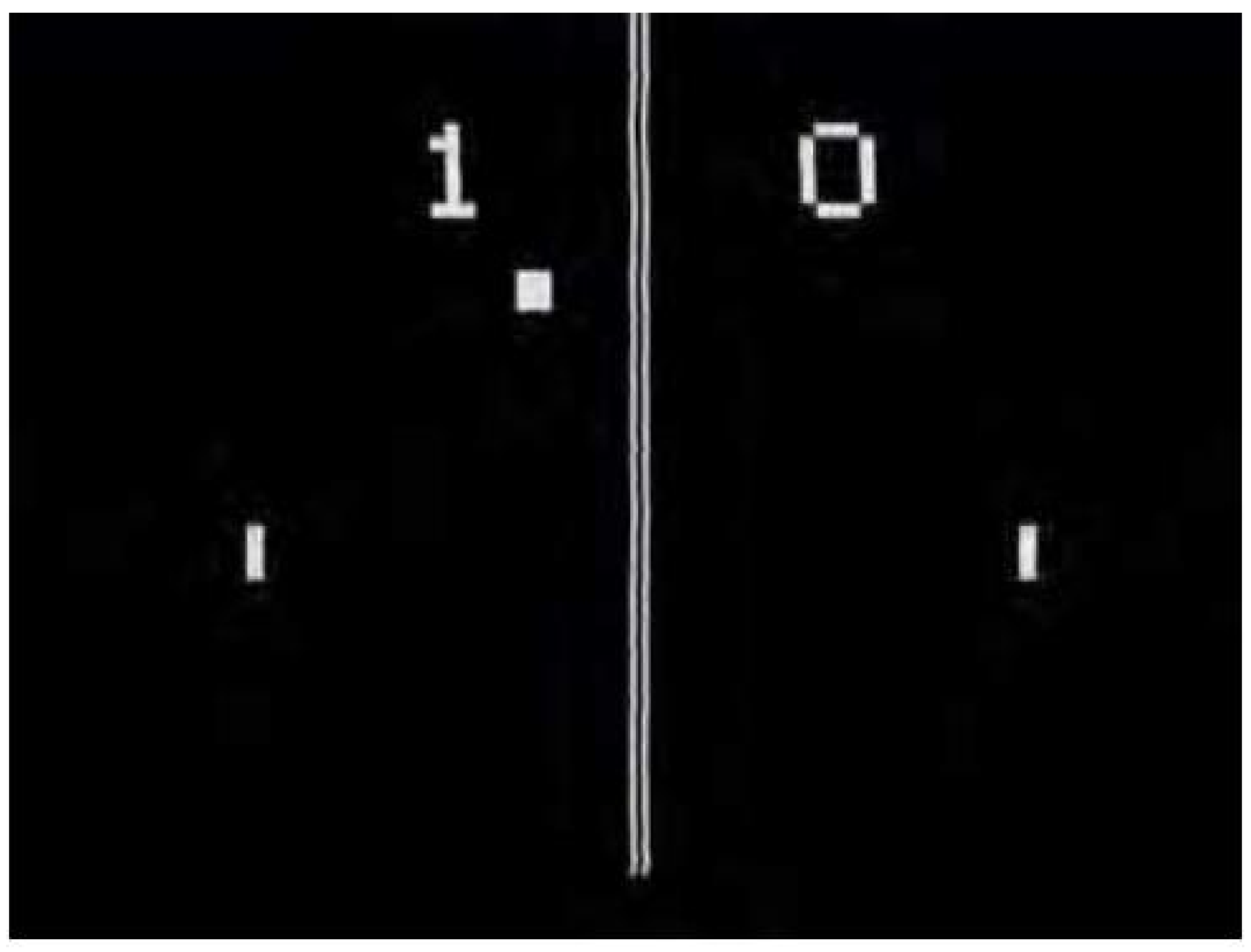
With that taken care of, the remainder of the visible lines can be neatly broken up into sections that can be handled sequentially.

From left to right, these are the Player 1 bat, Player 1 score, the net, Player 2 score and Player 2 bat. For the bats and net, we briefly toggle the polarity of the PWM signal, thus getting the XOR effect as the ball passes over, so the ball does not 'merge' with them.

Fig.2 shows how the horizontal lines are organised, and Screen 1 shows it without the lines.

The scores are handled slightly differently. These are effectively bitmaps hard-coded as brief assembly language sequences, so the PWM output is turned off while the scores are being displayed. Thus, the ball disappears behind the scores, which could provide an advantage for a canny player.

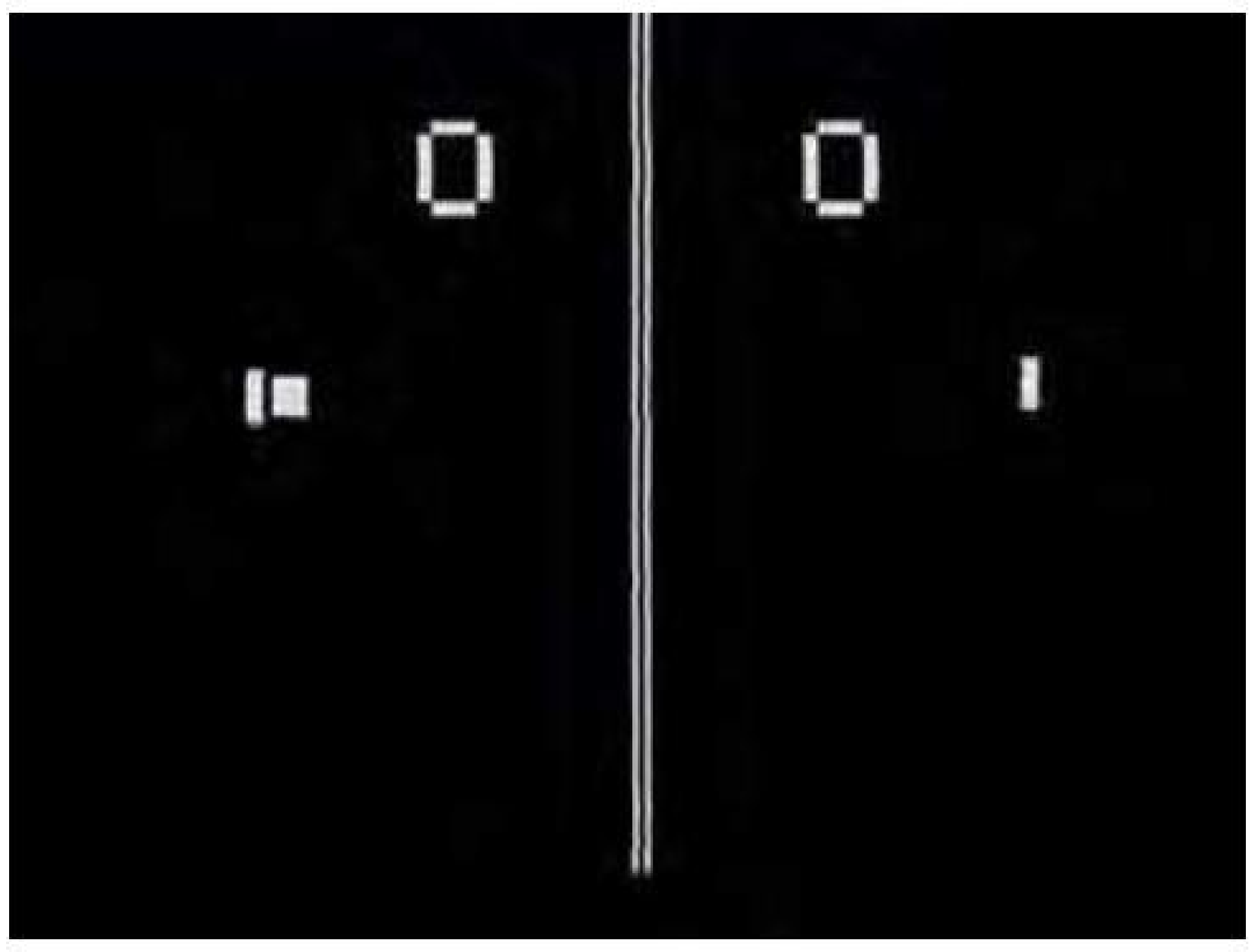
After the 256 active lines, a further 21 blank lines are displayed, followed by a single customised blank line that handles all of the logic that updates the game's state. The code for this line



Screen 1: a typical game of Nano Pong. The ball is in play after Player 1 has won the first point of the game.

SILICON
CHIP
NANO
PONG

Screen 2: with a reasonable amount of program Flash memory to spare, we added this splash screen when the unit is powered up.



Screen 3: the start of a game, before Player 1 has served the ball.

Parts List – Nano Pong

1 double-sided PCB coded 08105212, 43 x 16.5mm, from the PE PCB Service
1 PIC12F1572-I/SN (SOIC-8) programmed with 0810521B.HEX (IC1)
1 SMD mini Type-B USB socket (CON1)
1 10cm length of 20mm diameter clear heatshrink tubing
3 RCA plugs ***AND***
1 3m length of shielded cable ***OR***
1 triple RCA plug cable **[eg, Jaycar WV7316]**
1 5-way male pin header (CON2, **optional** for programming, see text)

Capacitors *(all 50V X7R SMD ceramic, M3216/1206-size)*
1 1µF 1 100nF

Resistors *(all 1% metal film SMD, M3216/1206-size)*
1 100kΩ (marked '104' or '1003')
1 10kΩ (marked '103' or '1002')
2 1kΩ (marked '102' or '1001')
6 470Ω (marked '471', '470R' or '4700')

Controller parts
2 small plastic enclosures (eg, UB5 Jiffy boxes)
2 1kΩ 24mm rotary potentiometers (VR1, VR2) **[eg, Jaycar RP3504]**
2 large knobs (up to 50mm) to suit potentiometers VR1 and VR2
2 momentary pushbuttons (S1,S2) **[eg, Jaycar SP0716]**
1 1m length (or longer) of 4-5 core wire for controllers **[eg, Jaycar WB1590]**
2 100mm cable ties

is two instructions shorter than the other blank lines, to account for the time taken to jump back to the start of the loop.

Most of the time that this final blank line is being generated, the game logic is processed. If it detects that the ball has struck a wall or bat, the ball vector is adjusted. This includes taking into account where it strikes the player's bat, as this affects the ball's vertical speed, like the original game. Also like the original game, each strike of the bat can also increase the ball speed.

These events also trigger a sound to be played, generated by a different PWM channel playing a tone from pin 5 until it is reset on the next field. This gives a variety of differently toned beeps depending on what the ball has struck.

The two ADC channels for the paddles are alternately sampled and allocated to their respective players. The relationship between the ADC value and on-screen position is adjusted to take into account the range set by the resistors.

If the ADC value is outside this range, then the bat position is not updated, which also takes care of the case when the ADC pin is pulled low by the button press. Thus, trick serves are not possible.

A point is registered whenever the ball reaches the screen edges (ie, missing the player's bat), which increments the score counter. Flags are set to indicate that the player winning the point is to serve, and if the score has reached 11, that a win has occurred.

In this case, a melody is played on the pin 5 PWM channel and the winning score is flashed. Timing for these events comes from different bits in the FIELDCOUNT parameter.

Since the main program only uses about 2/3 of the available Flash memory, we also added a splash screen, shown in Screen 2. This uses data from the score bitmap sequences in a hard-coded loop. An 8-bit timer counts down over 256 fields at 50Hz, so this screen shows for around five seconds.

Component notes

With such a small PCB, there isn't a lot that can be modified. If you find that the volume of the sounds doesn't match your other TV sources, you can adjust the 1kΩ/470Ω divider connected to pin 5 of IC1. Reduce the 470Ω part value (or increase the 1kΩ part value) to reduce the volume. Alternatively, increase the 470Ω part value to increase the volume.

If you want to make the game harder, you can change the 470Ω resistors connected to the potentiometers at pins 1 and 3 of CON4 and CON5. Increasing their value will create a gap near the top and bottom of the screen that the bats can't reach, as in the original Pong game.

For example, replacing these four 470Ω resistors with 560Ω resistors will leave around a 3% gap at the top and bottom of the bat travel.

If you have a 5V power source that doesn't have a USB connector, it can be fed to pins 2 (positive) and 3 (negative) of CON2. We haven't tested it, but the circuit should run from a 4.5V supply, such as three AA or AAA batteries in series.

Construction

In keeping with the theme of this being a modernised and miniaturised version of Pong, the PCB uses mainly SMD components. Since these are resistors and capacitors, with one IC in a relatively large 8-pin SOIC package, plus the USB socket, assembly is not difficult.

The double-sided PCB is coded 08105212, measures 43 x 16.5mm and is available from th *PE PCB Service*. Refer to the PCB overlay and wiring diagram (Fig.3) during construction.

Start by mounting the SMDs. We recommend that you have a temperature adjustable soldering iron, flux paste, tweezers, a magnifier and solder wicking braid, as well as some solder wire. The small PCB

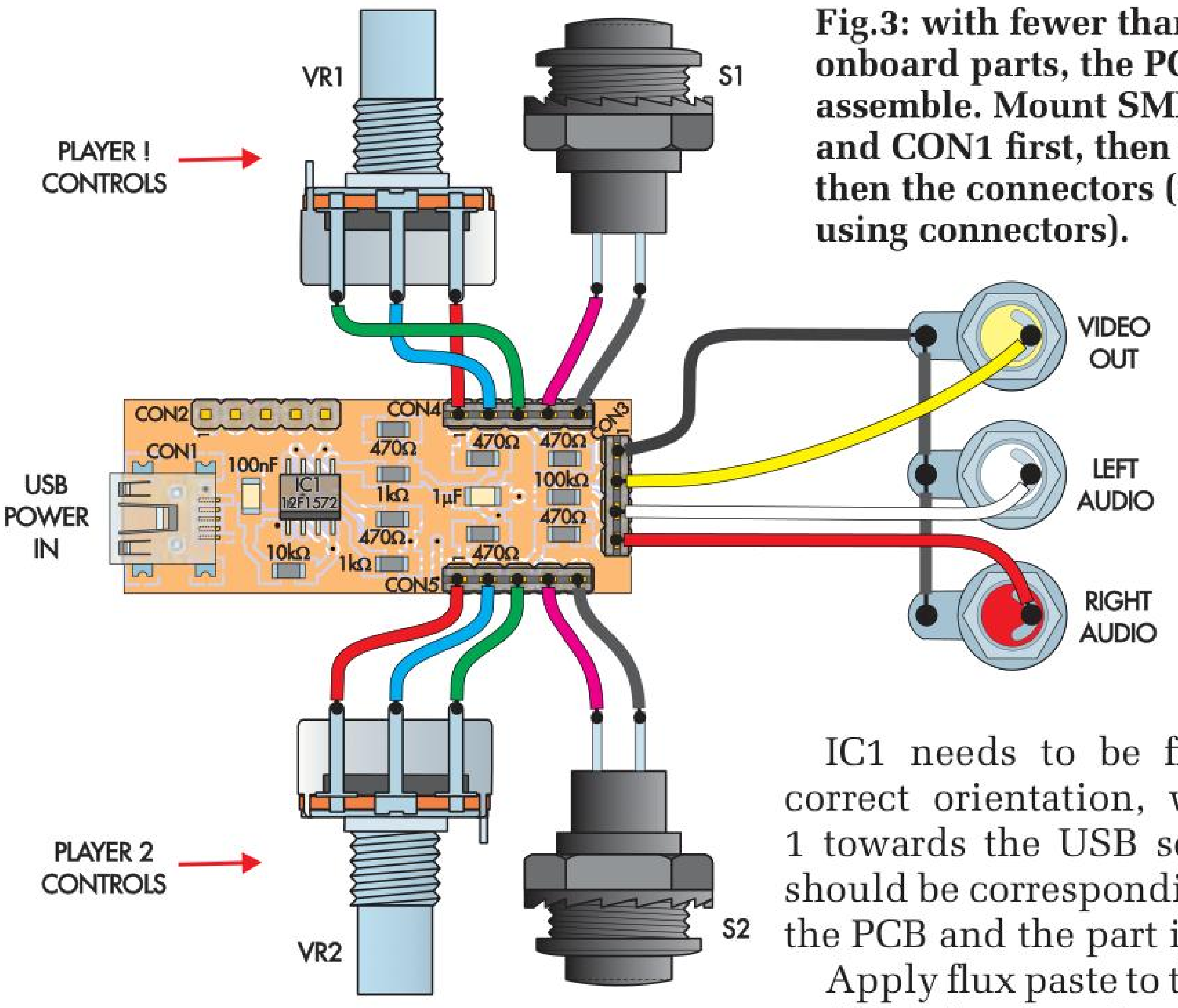


Fig.3: with fewer than 20 onboard parts, the PCB is easy to assemble. Mount SMD parts IC1 and CON1 first, then the passives, then the connectors (if you are using connectors).

can be temporarily secured to your desk with some Blu-Tak or similar material, so it doesn't move around during assembly.

Fume removal or ventilation is also recommended, as flux generates more smoke than typical solder wire.

Start by fitting IC1 and CON1. Apply flux to the pads and rest CON1 in place, then add a small amount of flux to the top of the pads. Its small plastic pegs should align it to holes in the PCB.

Clean the tip of the iron and add some fresh solder. Apply the iron's tip to the two longer pads on the PCB; the flux should help the solder run up the leads. You only need to solder the two outer leads as this socket only supplies power.

If you create a solder bridge, add some more flux and press the solder braid against the bridge until it draws up any excess solder. There should still be enough solder left to make a successful connection.

Turn up the iron temperature slightly to solder the four larger pads to the PCB that mechanically secure the connector, then return the iron to its original setting.

IC1 needs to be fitted in the correct orientation, with its pin 1 towards the USB socket. There should be corresponding marks on the PCB and the part itself.

Apply flux paste to the PCB pads and rest the IC in place. Add a little solder to the iron top and touch it to one pin to tack the part in place. If the IC isn't flat against the PCB or the pins are not aligned with their pads, carefully apply the iron again and adjust the position.

Once it is correctly aligned, solder the remaining pins. Then, if you have bridged pins, use the braid to remove them as described above.

The 100nF capacitor sits between IC1 and CON1. Using a similar technique to IC1, tack one lead, adjust and solder the other. Go back to the first lead and add a little flux paste or solder to freshen it up.

Don't be alarmed if your solder joints don't have compact, concave fillets. The important thing is that the parts are connected firmly; a large glossy solder joint that isn't bridging to other parts is fine.

Now fit the remaining SMD passives where shown in Fig.3. The resistors are usually marked (see the typical codes in the parts list), while the capacitors will only have their values printed on the packaging.

Once all the surface-mounted parts are fitted, you can clean the excess flux from the PCB using your preferred flux cleaning solution. Allow the board to dry out thoroughly before continuing.

All of the through-hole headers are optional, except CON2, which is required for programming the microcontroller. CON3-CON5 can be regular headers or sockets, or you can just solder wires (eg, sections of ribbon cable) directly to the PCB pads.

Programming IC1

You can program the chip after soldering it to the board, but it's a bit tricky. The problem is that the programming pins, pins 7 (ICSPDAT) and 6 (ICSPCLK) are also connected to the player paddle wipers. So we recomend you program the chip *before* connecting up those paddles.

To do this, plug your programmer into the ICSP header, CON2. You can solder a header strip to the pads, but we've had success by simply resting the header in place and applying gentle force to ensure contact.

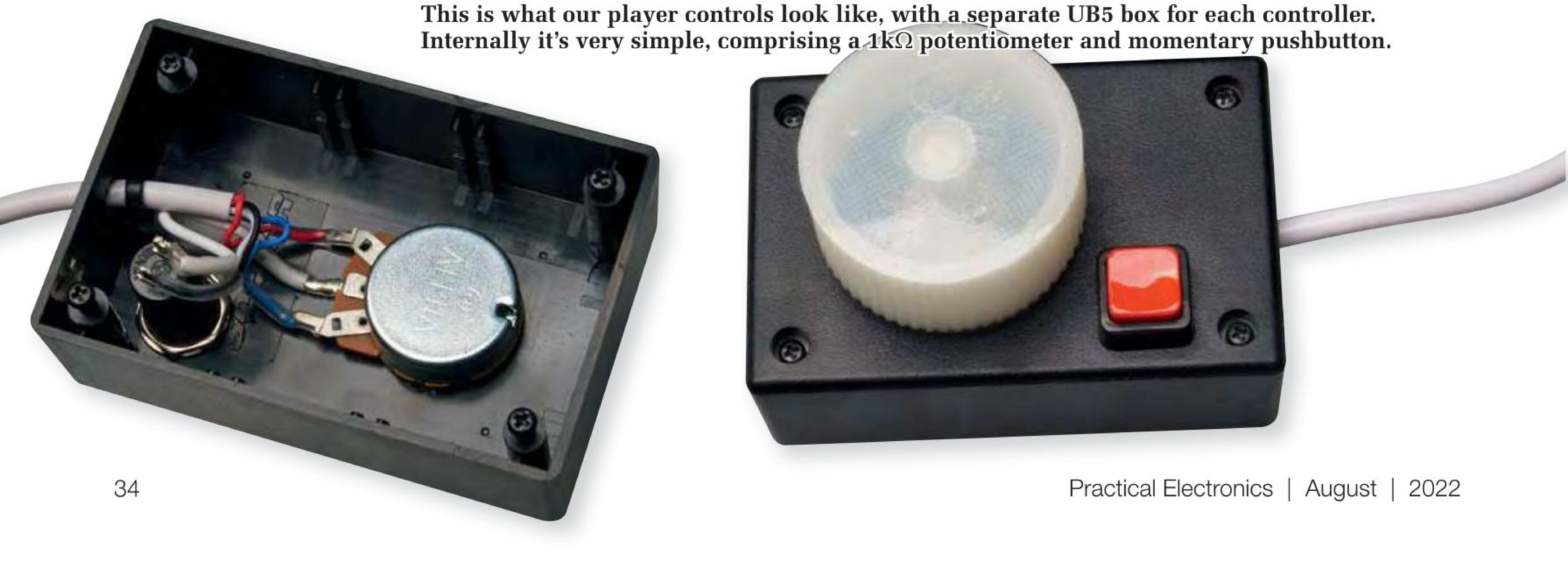
A PICkit 3 or PICkit 4 can be used, or even a Snap programmer, if you can supply power to the board (which the Snap cannot do by default). The easiest way to do this is using the mini-USB socket, CON1.

Use software like Microchip's MPLAB X IPE to upload the **0810521B.HEX** file to the chip – this file is available for download from the August 2022 page of the *PE* website. There's nothing obvious to indicate that the chip is working, apart from using the software option to verify that the file has been transferred correctly.

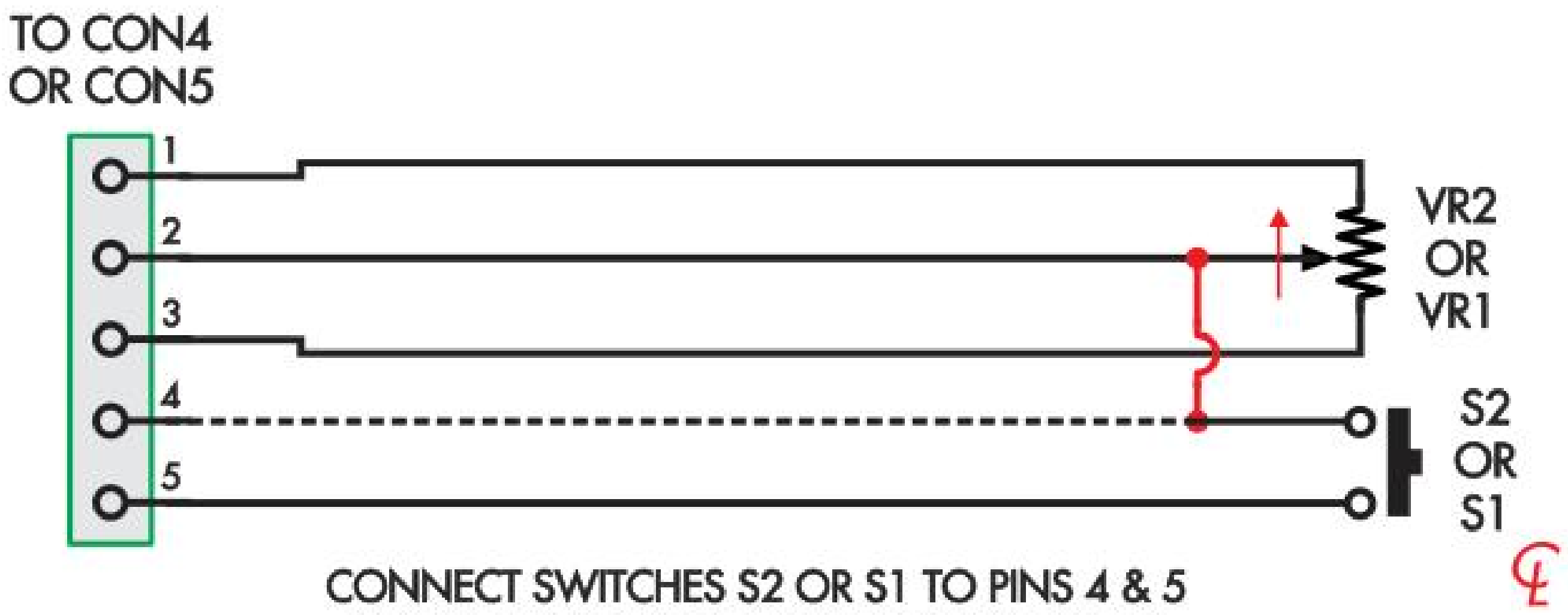
Wiring it up

We built the two player controls into plastic UB5 Jiffy boxes, but you could also mount all the parts in a single enclosure to imitate the hardware of the arcade version of Pong. Fig.4 shows the two possible ways that these can be wired.

The only difference is that if the controls are wired remotely, one end of the switch can be wired directly to

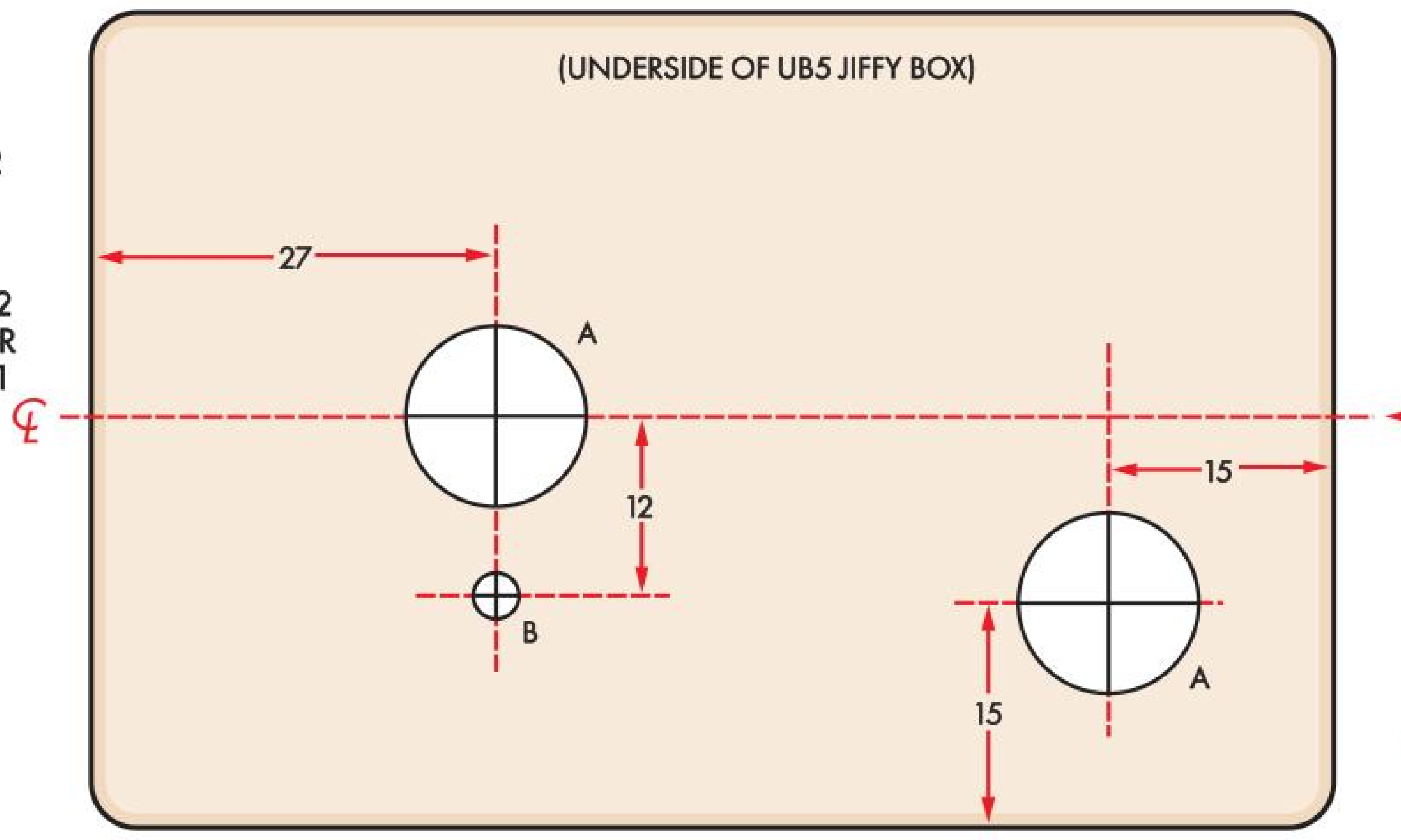


This is what our player controls look like, with a separate UB5 box for each controller. Internally it's very simple, comprising a 1kΩ potentiometer and momentary pushbutton.



CONNECT SWITCHES S2 OR S1 TO PINS 4 & 5 IF CONTROLS ARE INSIDE MAIN BOX, BUT IF THEY ARE IN SEPARATE BOXES CONNECT SWITCHES BETWEEN PINS 2 & 5 (AS SHOWN IN RED) TO ALLOW USE OF 4-WIRE CABLE.

Fig.4: wiring up the two player paddles/controllers externally only requires a four-core cable. Each paddle is wired the same, with Player 1 connecting to CON5 and Player 2 connecting to CON4. If everything is being mounted in the same enclosure, you can run the pot and switch wires back to CON4 and CON5 separately.



YOU WILL ALSO NEED A HOLE IN THIS END OF THE BOX TO SUIT THE CONNECTION LEAD

HOLES A ARE 12mm IN DIAMETER. HOLE B IS 3mm IN DIAMETER

ALL DIMENSIONS IN MILLIMETRES

Fig.5: we built our paddles into UB5 Jiffy boxes, with holes drilled in their bases as shown. There also needs to be a hole in the side of the box for the cable to pass through. Check that the size of your potentiometers, switches and wires match the hole sizes before drilling.

the pot wiper to save having to run an extra wire back to CON4 or CON5.

Fig.5 shows the cutting diagram that suits the parts we have used (listed in the parts list). You might need to modify the hole sizes if you are using different parts.

Our photos show how we have connected everything, but the design is quite flexible and can be adapted to different parts and enclosures. We'll describe how we finished our version.

Drill holes in a pair of UB5 Jiffy boxes according to Fig.5, noting that this should include a hole in the end of the box for the wire. The bottom of the box becomes the top when held in the hand. Doing it this way means that we aren't trying to juggle wires leading from the lid to the cable entry while mounting the lid.

Cut down the potentiometer shafts to suits the knobs you are using, and use a file to tidy up any rough corners or edges. This is most easily done with a hacksaw while holding the end of the potentiometer shaft (rather than the body) in a vice. This avoids straining the potentiometer mechanism.

Fit the pushbutton switches and potentiometers to the Jiffy boxes. You can also fit your knobs at this stage. We whipped up some 3D printed knobs to give a bit more grip. As with the firmware, the knob design files are available for download from the August 2022 page of the *PE* website.

Solder and heatshrink the wires as shown in Fig.3 and our photos. Run the connecting cable out through the hole and secure a cable tie around each cable to prevent it from being pulled off the terminals inside the box.

Next, solder the other ends of the wires to their respective pads on CON4 and CON5. If you want to test the paddle operation, apply power and check for 3.75V on the middle lead of each potentiometer in the fully clockwise position, and 1.25V anti-clockwise. This voltage should drop to 0V when the button is pressed.

There are NC (normally closed) variants of this switch, so if you find that the action is reversed, you might have this other variant.

To make the RCA connections for the TV, we simply cut a three-way RCA cable in half. Strip back a good amount of insulation and collect all the braids together. Attach these to pin 1 of CON3 (marked on the back with 'G'). We put a short length of heatshrink tubing over the braids for extra protection.

Then bare the internal wires by a small amount. The video plug (which will usually be yellow) should be connected to pin 2 of CON3, marked 'V'. Pins 3 should go to the left audio lead (white, 'L') lead, and pin 4 should go to the right audio lead (red, 'R').

Plug the RCA leads into the AV connections of a television and apply power to CON1. You should see the splash screen followed by the main game screen. Check that everything operates correctly.

If so, the main PCB can be sealed up by enclosing it in a length of 20mm diameter heatshrink tubing. Ensure that the CON1 end does not overhang the connector. A 10cm length should ensure that the cables are secure and have some strain relief.

Let's play!

At a game's start, the ball will be in front of one player's bat. Pressing the button on that controller will cause the ball to be served. Rotate the pot to move the bats on the screen to keep the ball in play.

If a player misses the ball, the other player wins a point. Once one player reaches eleven points, the game is over. The winning score will flash and a melody will play. Serving the ball starts a new game.

Reproduced by arrangement with
SILICON CHIP magazine 2022.
www.siliconchip.com.au

This is how we built our Nano Pong setup. It uses a separate controller for each player, and has a composite video connector.

Using Cheap Asian Electronic Modules

By Jim Rowe

USB Power Delivery Chargers

comsol
XY-PDS100
USB-A
505
V I-C I-A
TYPE-C

Left-to-right: the Comsol COWCC30WH, XY-PDS100 and Belkin F7U060AU

This article describes some low-cost modules that have appeared recently to take advantage of the dramatic growth in USB capability, especially in the area of power delivery (PD). This assortment includes PD chargers, cables and cable adaptors, while a follow-up article will look at 'trigger' or 'decoy' modules, used to configure the chargers, plus USB-PD testers.

When USB first appeared in the late 1990s, it could provide just 5V of power at up to 100mA for a 'low-power' device, or up to 500mA for a 'high-power' device, such as a USB hard disk drive.

But as the data transfer capabilities of USB were expanded via USB 2.0, USB 3.0 and finally USB-C, the power delivery capabilities were expanded as well.

USB 3.0 kept the 5V supply voltage but raised the 'high-power' current level to 900mA, allowing a downstream device to receive up to 4.5W (rather than just 2.5W).

When the USB-PD (Power Delivery) specification was finalised in 2012, a device could receive 5V at up to 1.5A or 7.5W of power via a standard Type-A to Type-B USB cable.

The smaller USB-C 24-pin connectors appeared in 2014, and when the USB-PD specification was further revised in 2014, 2016 and 2017, they increased the power delivery voltage and current levels as well.

Now devices can request power at either 5V, 9V, 12V, 15V or 20V, and can draw up to 5A – corresponding to 100W with a 20V supply. And since the USB-PD 3.0 revision of 2017, devices can also take advantage of the programmable power supply (PPS) protocol, which allows variation of the supply voltage in 20mV steps. This expands the possible USB-PD applications dramatically, and that's why we're seeing so many low-cost modules designed to take advantage of this increased flexibility.

How USB-PD works

The key idea to understand is that USB-PD is made possible thanks to extra contact pins in a USB-C connector. Specifically, the CC1 (A5) and CC2 (B5) pins, which are designated the Configuration Channel (CC) pins. The notional arrangement is shown in Fig.1.

Initially, a USB-PD-capable power supply sets its V_{BUS} output voltage to 5V. It also ties each of the CC pins of its output (downstream) USB-C connector to a logic high level via a pull-up resistor Rp, with the value of Rp chosen according to the supply's current capacity.

Devices designed to receive their power from the USB-C connector are fitted with a pull-down resistor Rd connected between one of the CC pins and ground. The value of Rd is chosen to indicate the current level wanted by the device.

As a result, when a cable from the device is plugged into the USB-C connector, the voltage drop on one of the CC lines indicates to the host that:

- ■ A load or 'sink' device is connected
- ■ The orientation of the USB-C plug in the connector
- ■ Current available from host supply.

There is then an exchange of data packets between the supply and the load/sink via the CC line, using DC-coupled

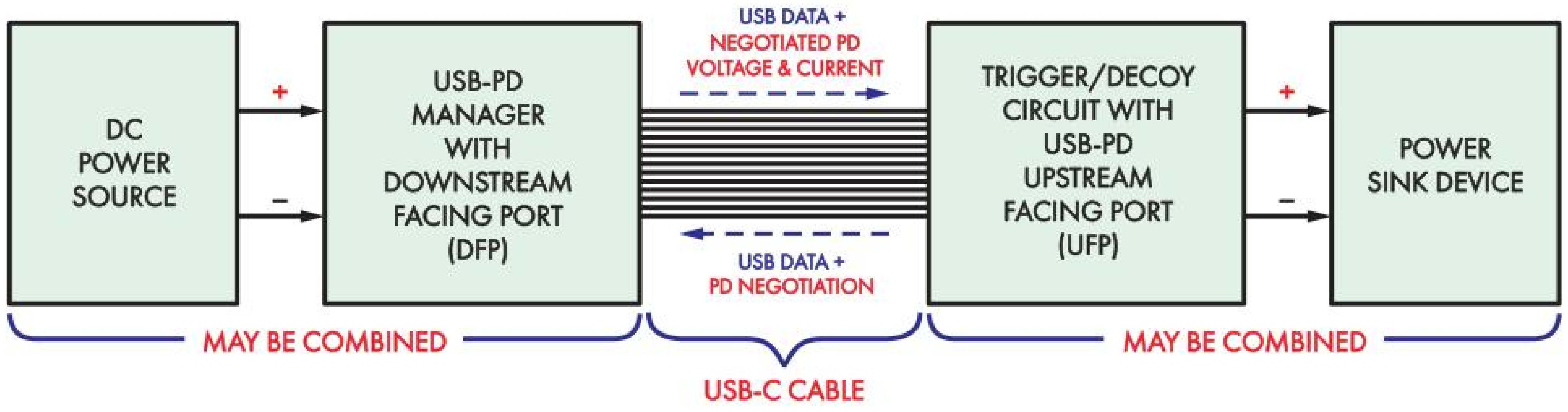

DC POWER SOURCE
+
–
USB-PD MANAGER WITH DOWNSTREAM FACING PORT (DFP)
USB DATA + NEGOTIATED PD VOLTAGE & CURRENT
USB DATA + PD NEGOTIATION
TRIGGER/DECOY CIRCUIT WITH USB-PD UPSTREAM FACING PORT (UFP)
+
–
POWER SINK DEVICE
MAY BE COMBINED
MAY BE COMBINED
USB-C CABLE

Fig.1: the USB-PD system consists of five elements: a primary DC power source, a USB-PD 'manager' with a downstream facing port (DFP), a USB-C cable, a trigger circuit fitted with an upstream facing port (UFP) and finally, the power 'sink'. The USB-PD manager element could be combined with the primary DC source, and the trigger circuit may also be combined with the sink.

BMC (Biphase Mark Code) or Differential Manchester encoding. This allows the load device to indicate the supply voltage it wants, and then the supply to change its output to the requested level if it can do so.

As mentioned above, if the supply supports the PPS protocol, the voltage can be adjusted in 20mV increments.

This negotiation can only occur if the load device is connected to the supply via a USB-C connector and matching cable. It won't work if a Type-A USB connector is used, because this lacks any CC pins or cable lines.

The initial USB-PD Rev.1 specification of 2012 allowed a device connected to a host/power supply via USB 2.0/3.0 Type-A and Type-B connectors to negotiate a higher voltage than 5V (eg, 12V or 20V) using a binary FSK signal on the V_{BUS} line. But this approach was deprecated when USB-PD Rev.2.0 was released in 2014.

So most USB-PD power supplies can only deliver 5V (or perhaps 12V) via their USB Type-A downstream port or ports.

Note that the USB-PD negotiation protocol allows for power to be transferred in either direction – from host to device or vice-versa. For example, a laptop or tablet PC can get its battery recharged quickly from a USB-PD power pack/charger by requesting that the charging be done at 9V, 15V or 20V instead of 5V.

The XY-PDS100 quick charger

This first module is a 'fast charger' that can be configured to give a range of output voltages and currents using the standard USB-PD protocol.

The XY-PDS100 comes in an extruded aluminium case measuring 53 x 46 x 21mm. It is available from several internet suppliers, including Banggood, which at the time of writing has it for £14 plus £2.50 for shipping.

As shown in the photos, the output end of the XY-PDS100 has a USB Type-A socket and a USB-C socket, plus a 3-digit 7-segment LED display (with 6.5mm-high digits) and three indicator LEDs. One lights when the output voltage is displayed, one when it's showing the current being drawn from the USB-C socket, and the third when showing the current drawn from the Type-A socket.

At the 'input' end, there are two sockets. One is a small concentric DC socket designed to accept 12-28V DC from a mains power supply, and the other a USB-C socket marked 'Input-PD'. On the underside of the case, the latter input has the legend 'PD Recommended 87W', but it seems to be simply an alternative DC input.

Essentially, what the XY-PDS100 does is convert a no-frills power supply with an output of 12-28V DC into a 'smart' USB-PD battery charger or power source, which can respond to the negotiation from a trigger unit to provide one of the standard charging voltage and current profiles.

So it's basically a programmable switch-mode step-down DC-to-DC converter, which can provide up to 100W of power at voltages between 5V and 20V from the USB-C output, or up to 36W of power at voltages between 5V and 12V from the USB Type-A output. And it even includes a three-digit LED readout displaying the current output voltage and current. Not bad for a very compact little unit that costs less than £20 delivered.

Because the XY-PDS100 is a step-down converter, it needs to have a DC input voltage at least 2V higher than the highest output voltage that could be requested. So if you only want a maximum of 12V for charging via the Type-A output, an input voltage of 14-15V would be fine. But for the full range of voltages required for USB-PD fast charging, the input voltage will need to be at least 22-23V.

I was quite happy with the measured performance of the XY-PDS100. It seems quite compatible with the PD 3.0 protocols, and also with the PPS 'vernier adjustment' protocol.

While the XY-PDS100 is a 'USB-PD Manager' module, needing an external DC supply, the remaining devices we're going to look at combine both functions, forming a complete USB-PD power source.

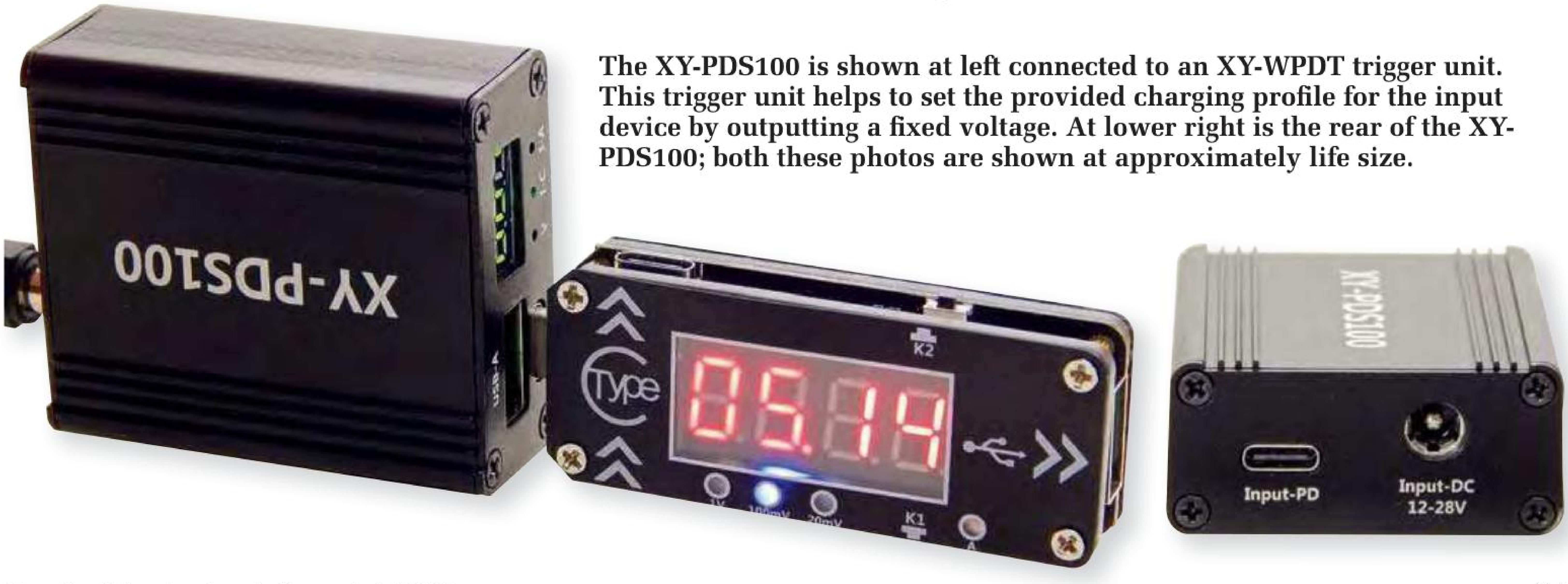
I had some difficulty obtaining them, though. I ordered a couple of units from a Chinese supplier, but they didn't arrive, and I eventually discovered that they were out of stock. So, I had to get them from local suppliers, but they cost significantly more than the units I had ordered from China. (Note that for me,

Using USB-PD for fast charging

Even before the USB-PD specification was released in 2012, various firms associated with the burgeoning mobile phone market worked out ways to use USB sockets for fast-charging mobile phone batteries. Examples are Qualcomm, which had developed its Quick Charge (QC) protocol, Motorola with its TurboPower protocol and Huawei with its SuperCharge (SC) protocol.

Perhaps because of the widespread application of these protocols, the various revisions of USB-PD gradually embodied them. As a result, when the USB-PD revision 3.0 was released in 2017, including PPS (Programmable Power Supply), it essentially incorporated just about all of the earlier fast-charging protocols.

So that's why the specifications of most of the USB-PD trigger modules and fast chargers will claim compatibility with a list of protocols such as PD 2.0, PD 3.0, Qualcomm QC3.0 and QC4+, Huawei SCP/FCP, Apple 2.4A, Samsung AFC, MediaTek PE2.0 and PE3.0, Oppo's VOOC and so on.



The XY-PDS100 is shown at left connected to an XY-WPDT trigger unit. This trigger unit helps to set the provided charging profile for the input device by outputting a fixed voltage. At lower right is the rear of the XY-PDS100; both these photos are shown at approximately life size.

Take care when buying USB-C cables and adaptors

Although you will find many low-cost USB-C cables from vendors on the internet, you need to be careful when buying many of them. For example, quite a few of the low-cost cables are really only suitable for providing power and battery charging, not transferring data, and especially not high-speed data transfer.

Apart from the lines involved in power transfer (including the configuration channel lines), they might not have any of the data transfer lines, except perhaps those for USB 2.0 (D+ and D–).

This applies particularly for cables fitted with a Type-A plug at one end and a USB-C plug at the other. In fact, the presence of a Type-A plug is a strong indication that a cable is not suitable for high-speed data transfer, and quite possibly only for power transfer and charging. And the power transfer/charging will only be possible at 5V, since negotiation of a higher supply voltage probably won't be possible.

This also applies to the many nominal USB-C adaptors. If these have a USB Type-A plug or socket at one end, that means they are probably only suitable for use in power transfer and charging, although they might be fine for low-speed and full-speed USB data transfer via the D+ and D– lines, assuming those wires are even fitted.

Even if a low-cost cable has USB-C connectors at both ends, that is no guarantee that it is suitable for really high-speed data transfer. This makes it a bit risky buying these cables via the Internet, because you can't test them before you buy them.

In fact, if you see one of these cables for less than £9, you can probably assume it's only suitable for power transfer and battery charging. USB-C cables capable of being used for really high-speed data transfer are likely to cost significantly more than that.

'local' means Australia, but I'm sure you can get similar if not identical versions in your part of the world.)

The Belkin F7U060AU 27W power adaptor

This unit cost around £20 from JB Hi-Fi (**www.jbhifi.com.au**). It measures just 51 x 60 x 31mm and weighs 50g. The unit is pictured in the rightmost photo at the start of this article; it has a two-pin mains plug on one end and a USB-C socket on the other end. That's it – it's just an elongated version of the familiar USB plugpack. The inscription on the plug end advises that it was designed in California and assembled in China.

When I tried it out with a couple of different trigger units, I found that although it would register as a PD 3.0 device, it would only provide a choice of three output voltages: 5V, 9V or 12V. The two lower voltage settings can provide up to 3A of current, while the 12V setting can provide up to 2.25A.

So the power rating of 27W only applies when the unit provides 9V or 12V; when it's providing 5V, it is really a 15W source. Of course, this would be fine if you only wanted up to 12V and 15-27W.

The Comsol COWCC30WH 30W wall charger

This unit also cost £20 from Officeworks (**www.officeworks.com.au/shop/**). It measures 44 x 64 x 40mm, and weighs 80g. As you can see from the leftmost photo at the start of this article, it's very similar to the Belkin unit, with a two-pin mains plug at one end and a USB-C socket at the other end. The inscription on its plug end simply says 'Made in China'.

When I checked this unit with a couple of different trigger units, it only registered as a PD 2.0 device, but could provide any of the full five output voltages: 5V, 9V, 12V, 15V or 20V. As with the Belkin unit, it could provide up to 3A at 5V or 9V, but at 12V, it could provide up to 2.5A. Then at 15V, it could provide up to 2A, while at 20V, it could provide up to 1.5A.

So it's only a 30W power source for three of the five selectable voltages.

Considering that its price is virtually the same as the Belkin unit, the fact that it provides a choice of the full five PD voltages, and with a nearly consistent power capability of 30W, makes it better value for money.

The range of voltages and currents available from this type of charger means that it could power a wide range of devices, including those you might build yourself.

If each of those devices contains circuitry to negotiate the current and voltage required, that means you could have a small selection of power supplies to power a wide range of devices.

So, in essence, these chargers could be the new 'multi-voltage plugpack' we all use in future.

The ALOGIC WCG1X65-ANZ 65W wall charger

The third USB-PD wall charger I bought is the ALOGIC WCG1X65, which again is very similar in size to

USB-C breakout boards

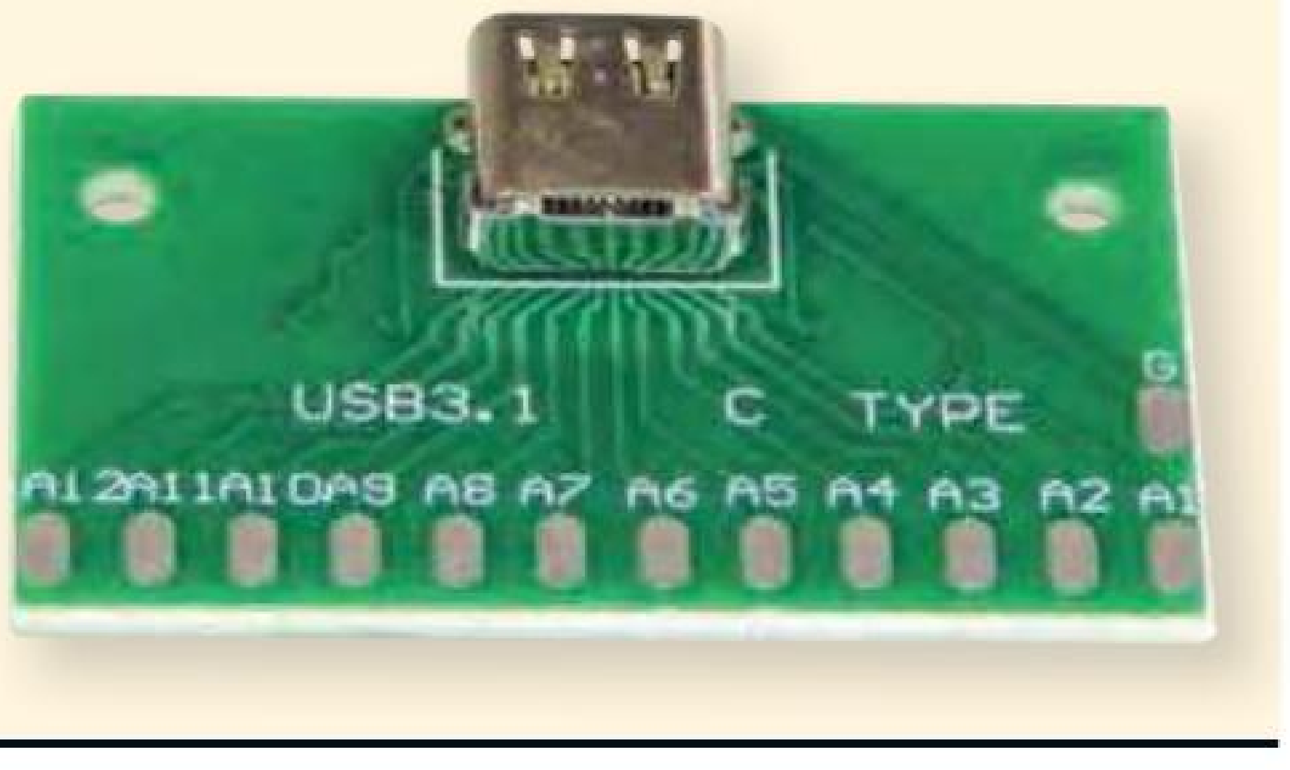
Because of the possible problems associated with USB-C cables, you might be interested in the low-cost 'breakout' module or test board shown in the photo below. It is available from internet suppliers like Banggood for only US$2.10 for a single, US$4.80 for a pack of five or US$9.00 for a pack of ten (all plus shipping, of US$3.30 in each case).

This module's PCB measures only 25 x 40mm and has a USB-C socket mounted at the centre of one of the 40mm sides. All 24 of the socket's connections are brought out to two rows of 12 solder pads at the opposite edge of the PCB, with one row (A1-12) on the top and the other (B1-12) underneath. The socket's metal frame is also brought out to a further 'G' pad on each side of the PCB.

A pair of these 'breakout' boards make it easy to test all of the lines and connections in a USB-C cable. I bought a pack of five, but wasn't too impressed with the soldering for the 24 very closely spaced pins of the sockets; one of them seemed to have a dry joint or two.

Since it would not be easy to repair these joints manually because of the very close spacing (about 0.5mm), I decided that the board concerned was throw-away material – so be warned!

In my next article, we'll take a look at some of the low-cost USB PD 'trigger' modules that can be used to set the output voltage and current of USB power supplies, like the ones described here.



the Belkin and Comsol units. It's slightly smaller, measuring 55 x 60 x 35mm, and weighs close to 95g. This unit also came from JB Hi-Fi, at a cost of around £40 plus delivery.

While it is almost twice the price of the other wall chargers, it boasts over twice the power capability at 65W. It comes with a 2m-long USB-C charging cable and a tiny (90 x 110mm) four-page quick start guide. It also features a white LED power indicator, just below the USB-C output socket.

PD
65W
0
1
2
3
4
5
6

The ALOGIC WCG1X65-ANZ 65W wall charger, shown enlarged for clarity. It registers as a PD 3.0 compliant device, and therefore can provide the standard voltages of 5V, 9V, 12V, 15V and 20V at 3A (or 3.25A for 20V). As the output power increases, these chargers can become quite costly.

When I checked this unit with the same trigger units as before, it registered as a PD 3.0 device and could easily be programmed to give any of the five standard PD voltages: 5V, 9V, 12V, 15V or 20V. And it can provide up to 3A at any of the four lower voltages, or up to 3.25A at 20V, which is pretty impressive considering its compact size and weight.

The makers claim that this is a result of using 'the latest GaN charging technology'. Presumably, they are taking advantage of the ability of transistors and diodes using gallium nitride (GaN) substrates to operate at much higher voltages and with higher efficiency.

So if you need a USB-PD wall charger capable of supplying up to 65W of power at any of the five PD 3.0 voltage levels, the ALOGIC WCG1X65-ANZ would be the best choice despite its significantly higher cost.

Note that one of the devices that I tried and failed to source from China was the Bakeey HC-652CA 65W wall charger, which would probably also be a good choice, if and when it becomes available.

Useful links

USB-C	**https://w.wiki/nto**
USB-PD	**https://w.wiki/34dT**
	https://bit.ly/pe-aug22-usbc
Quick Charge	**https://w.wiki/34dU**
Gallium nitride	**https://w.wiki/34dV**

Reproduced by arrangement with
SILICON CHIP magazine 2022.
www.siliconchip.com.au

Fluke/Philips PM3092 Oscilloscope
2+2 Channel 200MHz Delay TB,
Autoset etc – £250

STEWART OF READING
17A King Street, Mortimer, near Reading, RG7 3RS
Telephone: **0118 933 1111** Fax: **0118 933 2375**
USED ELECTRONIC TEST EQUIPMENT
Check website **www.stewart-of-reading.co.uk**
(ALL PRICES PLUS CARRIAGE & VAT)
Please check availability before ordering or calling in

HP 54600B Oscilloscope
Analogue/Digital Dual Trace 100MHz
Only £75, with accessories £125

LAMBDA GENESYS	PSU GEN100-15 100V 15A Boxed As New	£400
LAMBDA GENESYS	PSU GEN50-30 50V 30A	£400
IFR 2025	Signal Generator 9kHz – 2.51GHz Opt 04/11	£900
IFR 2948B	Communication Service Monitor Opts 03/25 Avionics	POA
IFR 6843	Microwave Systems Analyser 10MHz – 20GHz	POA
R&S APN62	Syn Function Generator 1Hz – 260kHz	£295
Agilent 8712ET	RF Network Analyser 300kHz – 1300MHz	POA
HP8903A/B	Audio Analyser	£750 – £950
HP8757D	Scaler Network Analyser	POA
HP3325A	Synthesised Function Generator	£195
HP3561A	Dynamic Signal Analyser	£650
HP6032A	PSU 0-60V 0-50A 1000W	£750
HP6622A	PSU 0-20V 4A Twice or 0-50V 2A Twice	£350
HP6624A	PSU 4 Outputs	£400
HP6632B	PSU 0-20V 0-5A	£195
HP6644A	PSU 0-60V 3.5A	£400
HP6654A	PSU 0-60V 0-9A	£500
HP8341A	Synthesised Sweep Generator 10MHz – 20GHz	£2,000
HP83630A	Synthesised Sweeper 10MHz – 26.5 GHz	POA
HP83624A	Synthesised Sweeper 2 – 20GHz	POA
HP8484A	Power Sensor 0.01-18GHz 3nW-10µW	£75
HP8560E	Spectrum Analyser Synthesised 30Hz – 2.9GHz	£1,750
HP8563A	Spectrum Analyser Synthesised 9kHz – 22GHz	£2,250
HP8566B	Spectrum Analsyer 100Hz – 22GHz	£1,200
HP8662A	RF Generator 10kHz – 1280MHz	£750
Marconi 2022E	Synthesised AM/FM Signal Generator 10kHz – 1.01GHz	£325
Marconi 2024	Synthesised Signal Generator 9kHz – 2.4GHz	£800
Marconi 2030	Synthesised Signal Generator 10kHz – 1.35GHz	£750
Marconi 2023A	Signal Generator 9kHz – 1.2GHz	£700
Marconi 2305	Modulation Meter	£250
Marconi 2440	Counter 20GHz	£295
Marconi 2945/A/B	Communications Test Set Various Options	POA
Marconi 2955	Radio Communications Test Set	£595
Marconi 2955A	Radio Communications Test Set	£725
Marconi 2955B	Radio Communications Test Set	£800
Marconi 6200	Microwave Test Set	£1,500
Marconi 6200A	Microwave Test Set 10MHz – 20GHz	£1,950
Marconi 6200B	Microwave Test Set	£2,300
Marconi 6960B	Power Meter with 6910 sensor	£295
Tektronix TDS3052B	Oscilloscope 500MHz 2.5GS/s	£1,250
Tektronix TDS3032	Oscilloscope 300MHz 2.5GS/s	£995
Tektronix TDS3012	Oscilloscope 2 Channel 100MHz 1.25GS/s	£450
Tektronix 2430A	Oscilloscope Dual Trace 150MHz 100MS/s	£350
Tektronix 2465B	Oscilloscope 4 Channel 400MHz	£600
Farnell AP60/50	PSU 0-60V 0-50A 1kW Switch Mode	£300
Farnell XA35/2T	PSU 0-35V 0-2A Twice Digital	£75
Farnell AP100-90	Power Supply 100V 90A	£900
Farnell LF1	Sine/Sq Oscillator 10Hz – 1MHz	£45
Racal 1991	Counter/Timer 160MHz 9 Digit	£150
Racal 2101	Counter 20GHz LED	£295
Racal 9300	True RMS Millivoltmeter 5Hz – 20MHz etc	£45
Racal 9300B	As 9300	£75
Solartron 7150/PLUS	6½ Digit DMM True RMS IEEE	£65/£75
Solatron 1253	Gain Phase Analyser 1mHz – 20kHz	£600
Solartron SI 1255	HF Frequency Response Analyser	POA
Tasakago TM035-2	PSU 0-35V 0-2A 2 Meters	£30
Thurlby PL320QMD	PSU 0-30V 0-2A Twice	£160 – £200
Thurlby TG210	Function Generator 0.002-2MHz TTL etc Kenwood Badged	£65

HP/Agilent HP 34401A Digital
Multimeter 6½ Digit £325 – £375

HP33120A	Function Generator 100 microHz – 15MHz	£350
HP53131A	Universal Counter 3GHz Boxed unused	£600
HP53131A	Universal Counter 225MHz	£350
Audio Precision	SYS2712 Audio Analyser – in original box	POA
Datron 4708	Autocal Multifunction Standard	POA
Druck DPI 515	Pressure Calibrator/Controller	£400
Datron 1081	Autocal Standards Multimeter	POA
ENI 325LA	RF Power Amplifier 250kHz – 150MHz 25W 50dB	POA
Keithley 228	Voltage/Current Source	POA
Time 9818	DC Current & Voltage Calibrator	POA

Marconi 2955B Radio
Communications Test Set – £800

Make it with Micromite

Phil Boyce – hands on with the mighty PIC-powered, BASIC microcontroller

Part 38: How to build a PicoMite BackPack

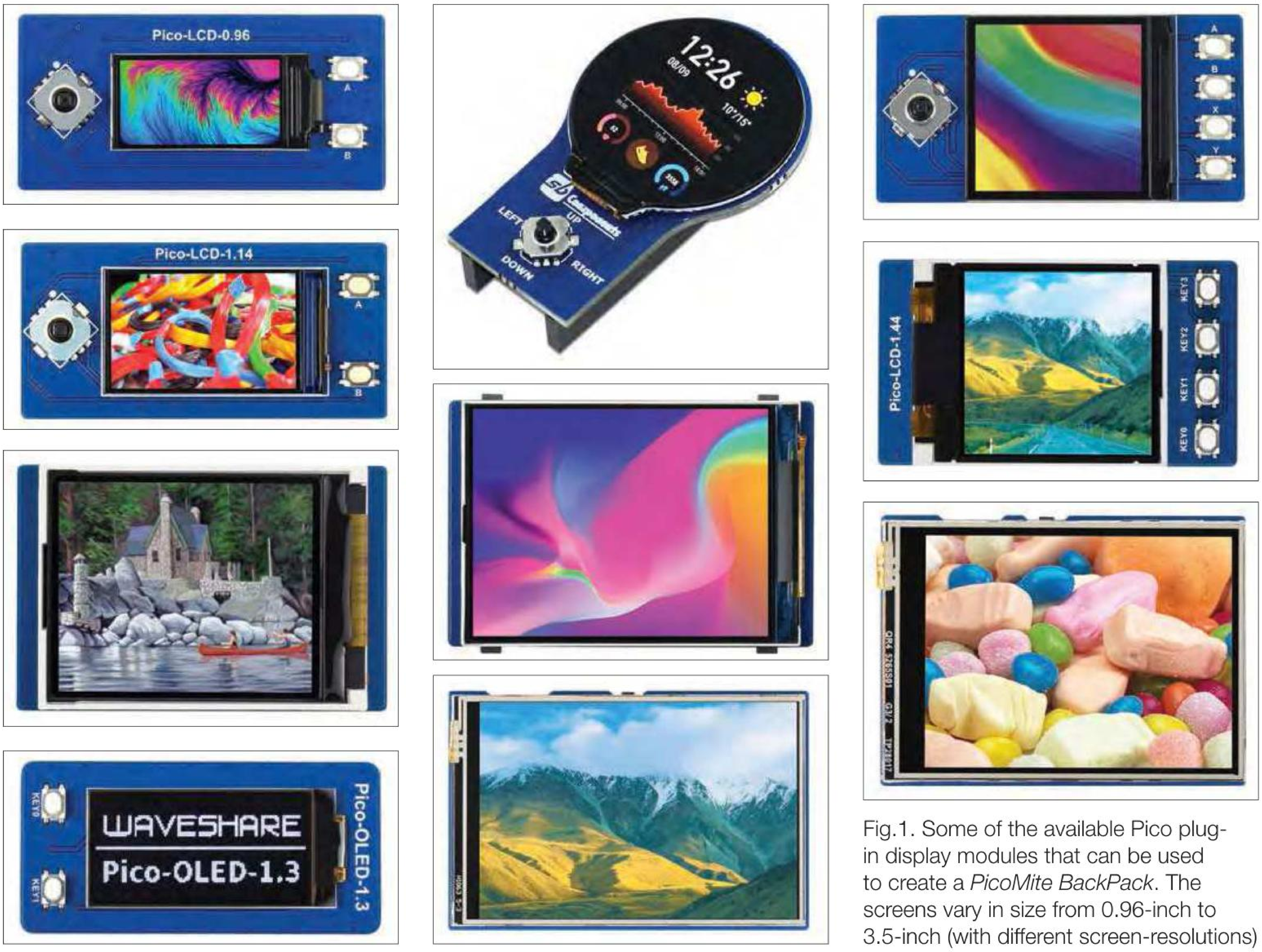


Fig.1. Some of the available Pico plug-in display modules that can be used to create a *PicoMite BackPack*. The screens vary in size from 0.96-inch to 3.5-inch (with different screen-resolutions) and come in a variety of technologies including standard LCD, high-clarity IPS and low-power OLED.

Last month, we showed how easy it is to load the MMBASIC firmware into a £4 Raspberry Pi Pico. The result – a 'PicoMite' – is an extremely powerful module that offers 26 I/O pins. Since it is based on the Micromite Plus firmware, it has all the features of a MicroMite Plus, which means there are many additional graphics commands available to the user compared to a standard Micromite. Furthermore, there are also commands that allow connection to an SD card, enabling storage of program, data, image and sound files. However, to be able to take advantage of these features we need to add a suitable display and SD socket to the PicoMite. This is where we can be smart and take advantage of the many readily available low-cost display modules that are designed to be plugged directly onto a Raspberry Pi Pico. By connecting an appropriate display module (preferably with an on-

Micromite code

The code in this article is available for download from the *PE* website.

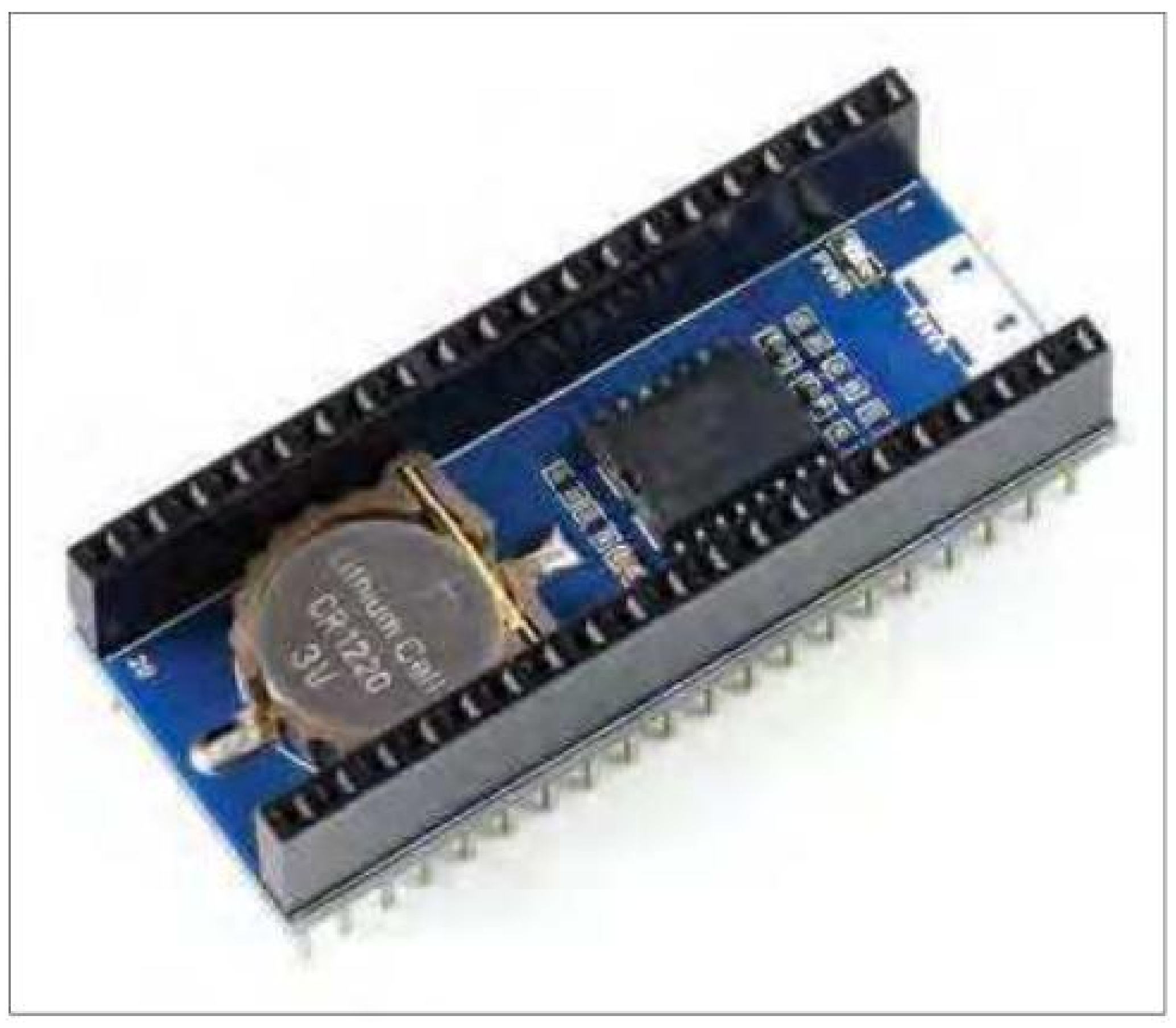


Fig.2. The Waveshare Pico-RTC-DS3231 is a high-accuracy Real-Time-Clock module; perfect for adding to a *PicoMite BackPack*. It has pins and sockets so it can easily be sandwiched between the PicoMite and the display module.

board SD socket) we effectively create a *PicoMite BackPack*. As we'll see, not only is the *PicoMite BackPack* much more powerful than the standard *Micromite BackPack*, but also it is much cheaper, and because no soldering is required, it is also much quicker to get a project up and running.

A point worth noting here is that many of the available Pico display modules are based on IPS screen technology, which means superior image quality – higher contrast and a wider viewing angle than the screens commonly used with the standard *Micromite BackPack*.

As a bonus, there is nothing stopping us from also using many of the other types of plug-in Pico modules with our PicoMite (ie, non-display modules). This is a great way to quickly (and often very cheaply) add a variety of other features to the PicoMite. For example, there is a high-accuracy Real-time Clock (RTC) module available based on our favourite RTC chip, the DS3231 (see Fig.2). An RTC is a useful addition to the *PicoMite BackPack*, so later in this article we'll show you just how easy it is to add one.

Display module choice

There are many different plug-in display modules available for the Pico (see Fig.1). The three fundamental parameters determining display choice are the physical screen size, the shape and the pixel resolution. Many of these displays can work directly with the PicoMite thanks to MMBASIC having various built-in display drivers, but do remember that as with any other Micromite, only one display can be connected to a PicoMite at any one time.

To create a *PicoMite BackPack*, all you need to do is choose a display for your project, connect it to the PicoMite, and then set a few `OPTION`s at MMBASIC's command prompt (with the parameters dependent upon the display module chosen).

To demonstrate this, we will discuss how to set up two different *PicoMite BackPacks*; one with a 2.8-inch 320x240 touchscreen (identical size and resolution to a standard *Micromite BackPack*) and the other with a 3.5-inch 480x320 touchscreen. Note that the process we use to create a *BackPack* applies to any of the compatible display modules; it is just the `OPTION`s parameters that vary.

Latest PicoMite Firmware

Since last month's article, a new version of the MMBASIC Firmware (v5.07.04) has been released. The required file (**PicoMiteV5.07.04.uf2**) is available for free download from the August 2022 page of the *PE* website (along with the updated *PicoMite User Manual*). We advise you to keep your PicoMite updated with the latest version of MMBASIC as this means you will have access to any new commands or features that are introduced, plus any bug-fixes.

An alternative source for PicoMite firmware updates is via Geoff Graham's website (**www.geoffg.net/picomite.html**). If you scroll to the Downloads section near the bottom of the page, you will find both files there, along with a text file containing a list of known bugs. You can also find the beta version of the next PicoMite firmware (currently v5.07.05b9). It is the most stable version of any beta software I have ever come across – credit to Geoff Graham and Peter Mather!

So, update your PicoMite to ensure you're running at least MMBASIC v5.07.04 (or v5.07.05b9). Remember that if you already have MMBASIC installed on your Pico module, then you can use the MMBASIC command `UPDATE FIRMWARE` to bring up the Pico's Flash drive (where you place the relevant **.uf2** file). However, if it is a brand new (blank) Pico module, then hold down the 'BOOTSEL' button while powering up the Pico – see last month in *Part 37*.

Pico Expander module

Before we begin creating our first *BackPack*, we want to alert you to a type of module that is useful when experimenting with a PicoMite. This is the 'Expander' module which allows multiple Pico-compatible plug-in modules to be used simultaneously by extending all the Raspberry Pi Pico's pins to multiple sets of pin headers (the Pico is plugged into one socket header set). Expander modules typically allow two or four plug-in Pico modules to be used – see Fig.3. Regardless of size (and shape), these Expander modules are also very useful as they provide quick and easy access to all I/O pins. By using the popular Dupont style jumper leads (with a female connector at one end for plugging onto the relevant Expander pin), the other end of the lead (if male) can then be inserted into a breadboard, or alternatively (if male or female) the other end can be attached to a socket or pin on another non-Pico module (as we will see later when using a low-cost, non-Pico RTC module).

So, even though an Expander module is not required to make our *BackPack*, do get one. It will be useful when you tinker with your new PicoMite projects. Make sure you also have a selection of Dupont leads (female-male, and female-female). We prefer leads that are approximately 20cm in length; and have never suffered any issues with peripherals failing to work due to impacted signal timing.

Now that we've updated the firmware, let's create a *BackPack*.

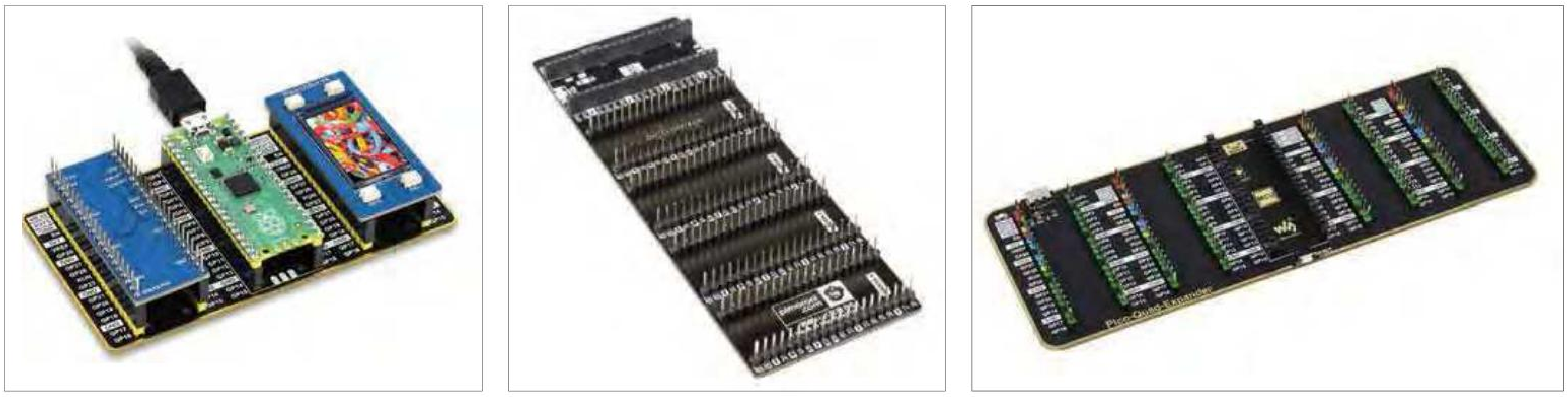


Fig.3. Pico Expander modules are available from different suppliers. Typically, they can connect either two or four Pico plug-in modules to the Pico. They also provide easy access to all I/O pins, which are clearly marked on the silkscreen.

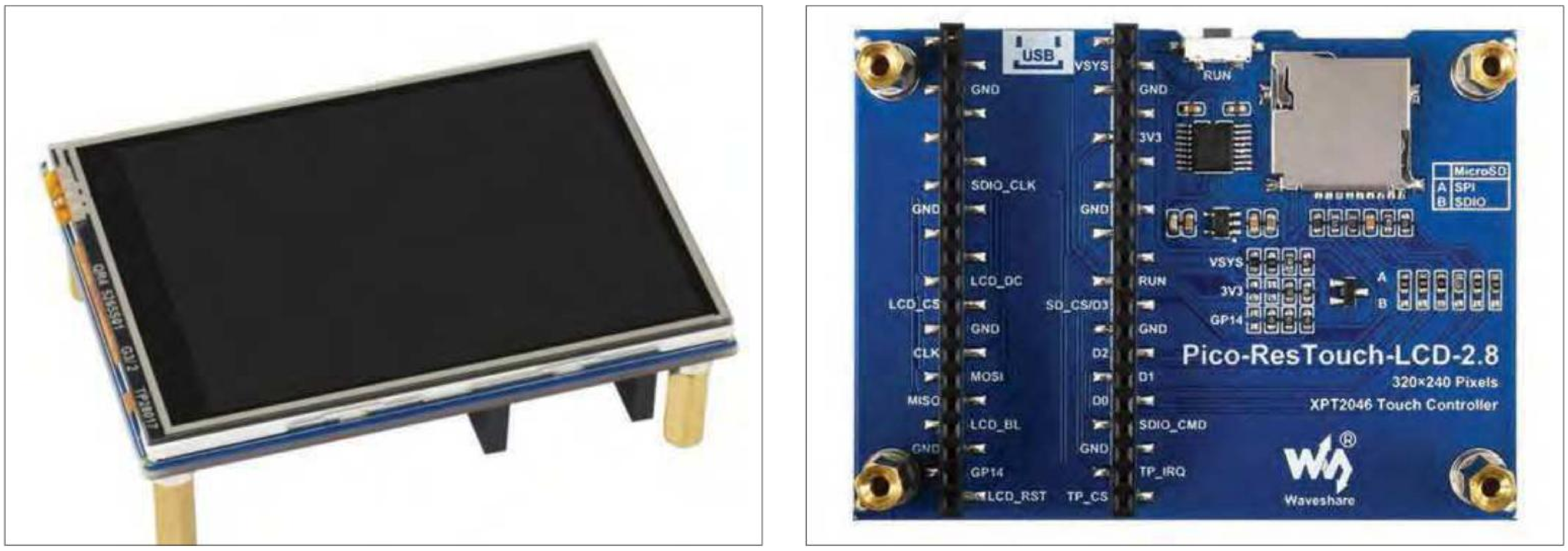


Fig.4. The Waveshare Pico-ResTouch-LCD-2.8 display module incorporates a 2.8-inch IPS touchscreen with a resolution of 320x240. It also an on-board micro-SD socket (perfect for making a PicoMite BackPack).

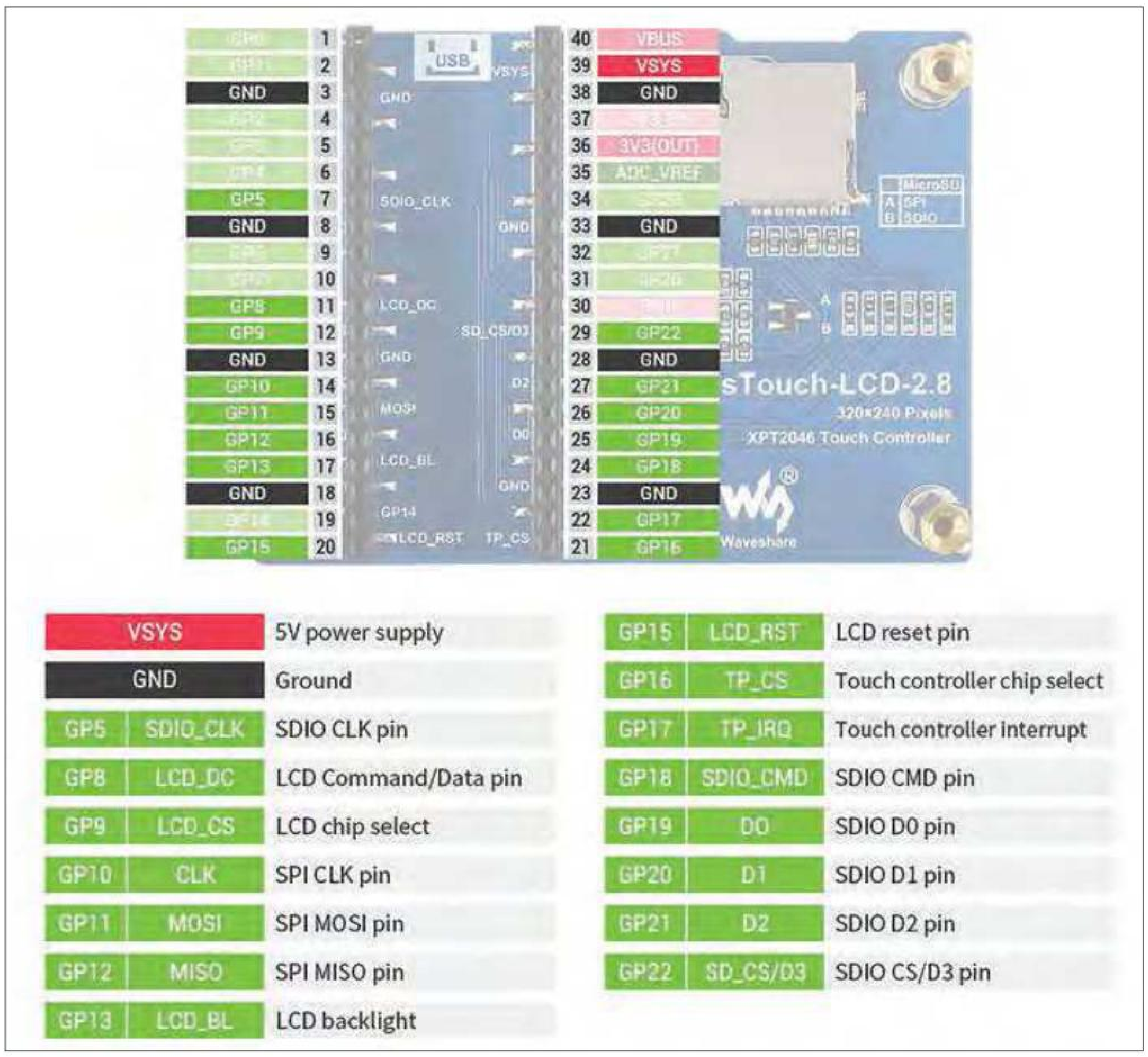
2.8-inch (320x240) BackPack

What we will create is an alternative to the standard 2.8-inch *Micromite BackPack*. We will use a display module from the popular online manufacturer Waveshare. The specific module is the Pico-ResTouch-LCD-2.8 (see Fig.4). If you search online for this exact model, you will soon find a supplier that has them in stock; typical cost is around £14.50 plus delivery. For readers who are interested in a detailed description of the module, visit: **www.waveshare.com/wiki/Pico-ResTouch-LCD-2.8**

The only information that we really need to understand from the datasheet is the pin-out information – see Fig.5. The reason for showing it here is that it will help us to define the correct parameter values when setting `OPTION`s.

Now connect this specific display module to your PicoMite, either by plugging the PicoMite directly into the display (Fig.6a), or by using an Expander module (Fig.6b).

Next, connect your PicoMite to your Terminal Application (ie, TeraTerm), and at the command prompt, work through the 12 steps shown below. (Note that the PicoMite is reset after setting an `OPTION` that saves parameters to memory, so you may need to press the `Enter` key to get the command prompt to appear again after the PicoMite has reset).



VSYS		5V power supply
GND		Ground
GP5	SDIO_CLK	SDIO CLK pin
GP8	LCD_DC	LCD Command/Data pin
GP9	LCD_CS	LCD chip select
GP10	CLK	SPI CLK pin
GP11	MOSI	SPI MOSI pin
GP12	MISO	SPI MISO pin
GP13	LCD_BL	LCD backlight
GP15	LCD_RST	LCD reset pin
GP16	TP_CS	Touch controller chip select
GP17	TP_IRQ	Touch controller interrupt
GP18	SDIO_CMD	SDIO CMD pin
GP19	D0	SDIO D0 pin
GP20	D1	SDIO D1 pin
GP21	D2	SDIO D2 pin
GP22	SD_CS/D3	SDIO CS/D3 pin

Fig.5. The pin functionality and numbering used on the Pico-ResTouch-LCD-2.8. This information is used to define what parameter values need to be entered in the various `OPTION`s that require setting.

1. Start by clearing any exiting `OPTION`s that have been set by typing the command: **`OPTION RESET`**
2. To check there are currently no `OPTION`s set (or have just been cleared from step 1), type the command: **`OPTION LIST`** (this should return no `OPTION`s – if you see any listed, simply type `OPTION RESET` again).
3. Set the PicoMite to run at its current maximum speed of 252MHz (this is not a requirement for the display module, but it is worth doing so that your PicoMite can perform optimally). To do this, type: **`OPTION CPUSPEED 252000`**
4. Next, we need to define the pins used for the SPI bus (the bus used to communicate with the display). We do this by using the command: `OPTION SYSTEM SPI` *`clk_pin`*, *`mosi_pin`*, *`miso_pin`* (replace the three parameters shown in italics with the relevant pin numbers). Fig.5 shows this is entered as: **`OPTION SYSTEM SPI GP10, GP11, GP12`**
5. To set the display driver, we use the command: `OPTION LCDPANEL` *`controller`*, *`orientation`*, *`DC`*, *`RESET`*, *`CS`*, *`backlight`* (replace the parameters with the relevant values). The controller used in this specific display module is the ST7789 (not the more familiar ILI9341). Refer to the *PicoMite User Manual* and you will see that there are various displays that use this controller (each with a different pixel resolution). To distinguish between the different ST7789 display modules, we add a reference to the horizontal screen resolution, so the *`controller`* parameter we need to use becomes ST7789_320. We will use Landscape (L) *`orientation`*, although we

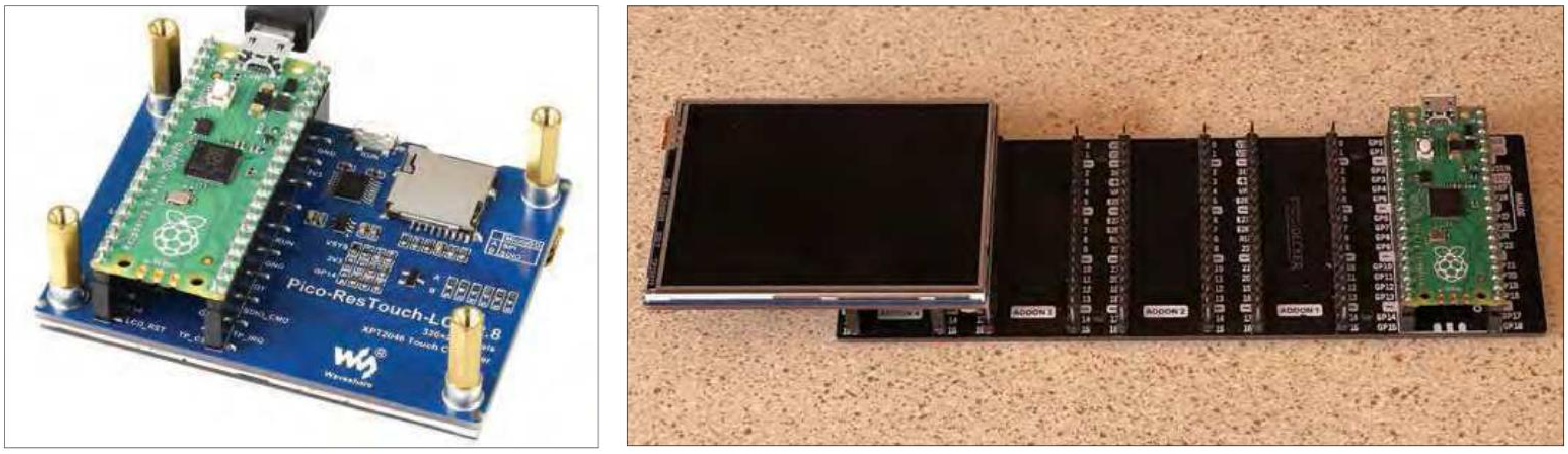
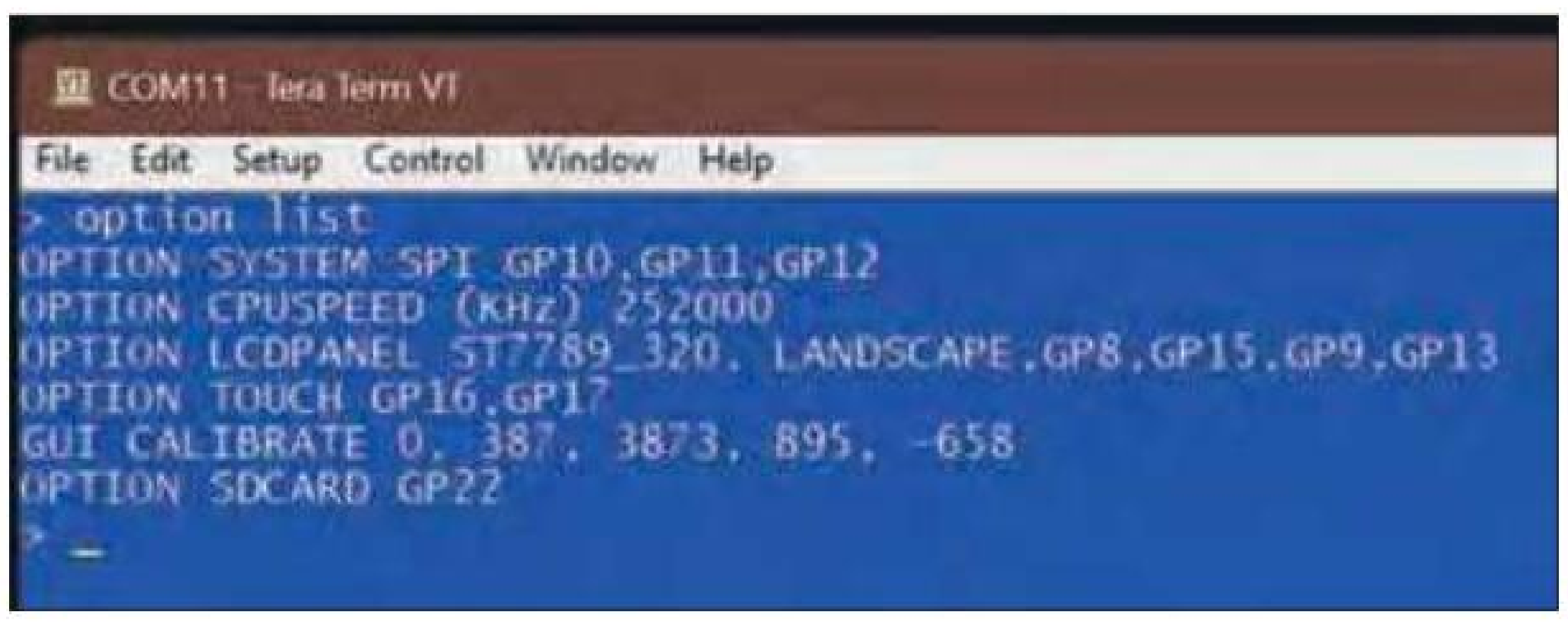


Fig.6. a) (left) Here, the 2.8-inch display module is shown attached directly to the PicoMite, resulting in a compact *PicoMite BackPack*. Ensure the PicoMite is oriented correctly (note: the display module's silkscreen highlights which end to align with the Pico's USB socket). Fig.6. b) (right) Alternatively, the display module and PicoMite can be connected together using an Expander module.



```
> option list
OPTION SYSTEM SPI GP10,GP11,GP12
OPTION CPUSPEED (KHz) 252000
OPTION LCDPANEL ST7789_320, LANDSCAPE,GP8,GP15,GP9,GP13
OPTION TOUCH GP16,GP17
GUI CALIBRATE 0, 387, 3873, 895, -658
OPTION SDCARD GP22
> _
```

Fig.7. Once the PicoMite's `OPTION`s have been set, it is worth typing `OPTION LIST` to check all parameters have been set correctly (here, the 2.8-inch settings are shown).

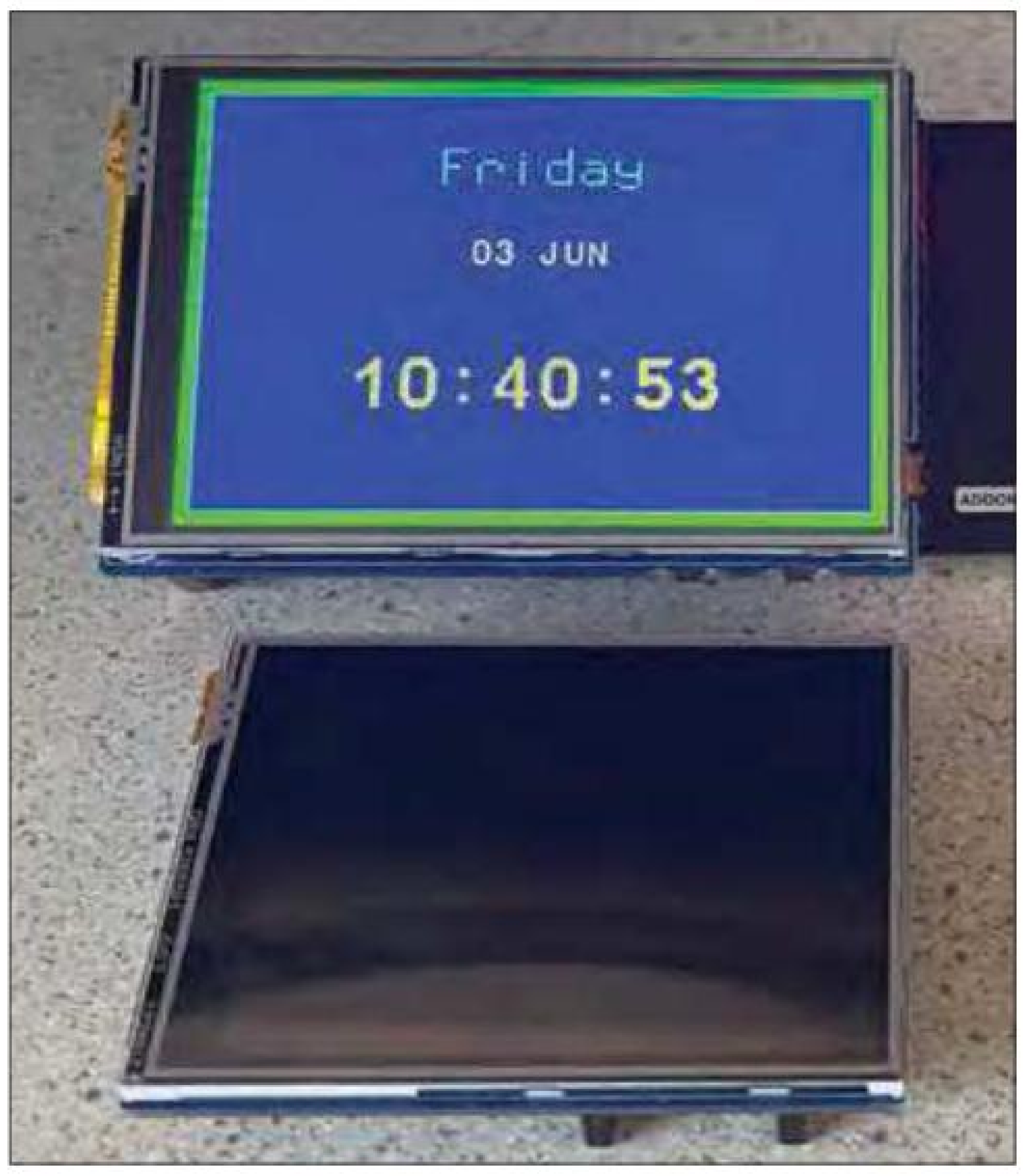
can choose portrait (P), reverse-landscape (rl) or reverse-portrait (rp). Therefore, with reference to Fig.5, we set the display driver with: **`OPTION LCDPANEL ST7789_320, L, GP8, GP15, GP9, GP13`**

6. To check the screen is set up correctly, use the command: **`GUI TEST LCDPANEL`** (which displays the usual test animation of randomly placed, randomly sized, coloured circles). Press any key to stop it. If you don't see this, then use `OPTION LIST` to display the `OPTION`s that are set, and carefully check all of the parameter values. If you spot an error, simply type: `OPTION LCDPANEL DISABLE` to delete the `OPTION`, and re-enter it again with the correct values.
7. To set up the touchscreen, we need to use the command: `OPTION TOUCH` *`T-CS, T_IRQ [,beep]`*. The optional *`beep`* parameter is not used here as the display module does not have an onboard beeper (you could add a beeper to any available I/O pin, and define this pin number as the *`beep`* parameter). Again, with reference to Fig.5, the command is entered as: **`OPTION TOUCH GP16, GP17`**
8. Next, calibrate the touchscreen with the command: **`GUI CALIBRATE`**. Simply follow the on-screen instructions – touch each corner in turn with a stylus (plastic gives best accuracy and least wear to the screen's touch-overlay). If calibration is performed successfully, minimal deviation will be reported, but, if necessary, repeat this command until you have near to zero deviation. If this sounds complex, don't worry, simply touch the crosshairs that appear on the screen, and then check the result displayed after all four corners have been touched. If either the X or Y value returned is higher than 5, then calibrate again. With a stylus and a steady hand, you can get zero deviation.
9. To check the touchscreen calibration, use the command: **`GUI TEST TOUCH`** and then use a stylus to draw on the screen. You should see the pixels turning on at the point of contact – if not, go back and repeat step 8. Press any key to exit this touch-test.
10. Next we will set up the SD socket by using the command: `OPTION SDCARD` *`SD_CS[, CLK, MOSI, MISO]`*. For this display module, we only need to define the *`SD_CS`* pin number, hence with reference to Fig.5, this is entered as: **`OPTION SDCARD GP22`**
11. To check the PicoMite can read an SD card, carefully insert a micro-SD card into the socket which is mounted on the back of the display module. Use an SD card that has between 4GB and 32GB capacity, and formatted in either FAT16 or FAT32. Then type the command: **`FILES`** to display a list of any files and/or folders that are on the SD card. If you see the message 'SD Card not found', then chances are you have entered an incorrect parameter value (or not fully inserted the micro-SD card). Check the parameter value by using `OPTION LIST`, and if necessary, use `OPTION SDCARD DISABLE` to delete the `OPTION`, and then repeat step 10 to re-enter it correctly.
12. As a final check, if you type **`OPTION LIST`** you should see exactly the same `OPTION` parameter values as shown in Fig.7 (but probably five different parameter values shown against `GUI CALIBRATE`)

Once you have successfully followed the 12 steps above, congratulate yourself; you have just created a working 2.8-inch *PicoMite BackPack*.

3.5-inch (480x320) BackPack

We will now describe how to set up a *PicoMite BackPack* that uses a bigger 3.5-inch touchscreen, and which has a higher pixel resolution. For this, we will use another display module from Waveshare, the Pico-ResTouch-LCD-3.5 (see Fig.8). Note that the typical price for the 3.5-inch display module is £18.50 (so



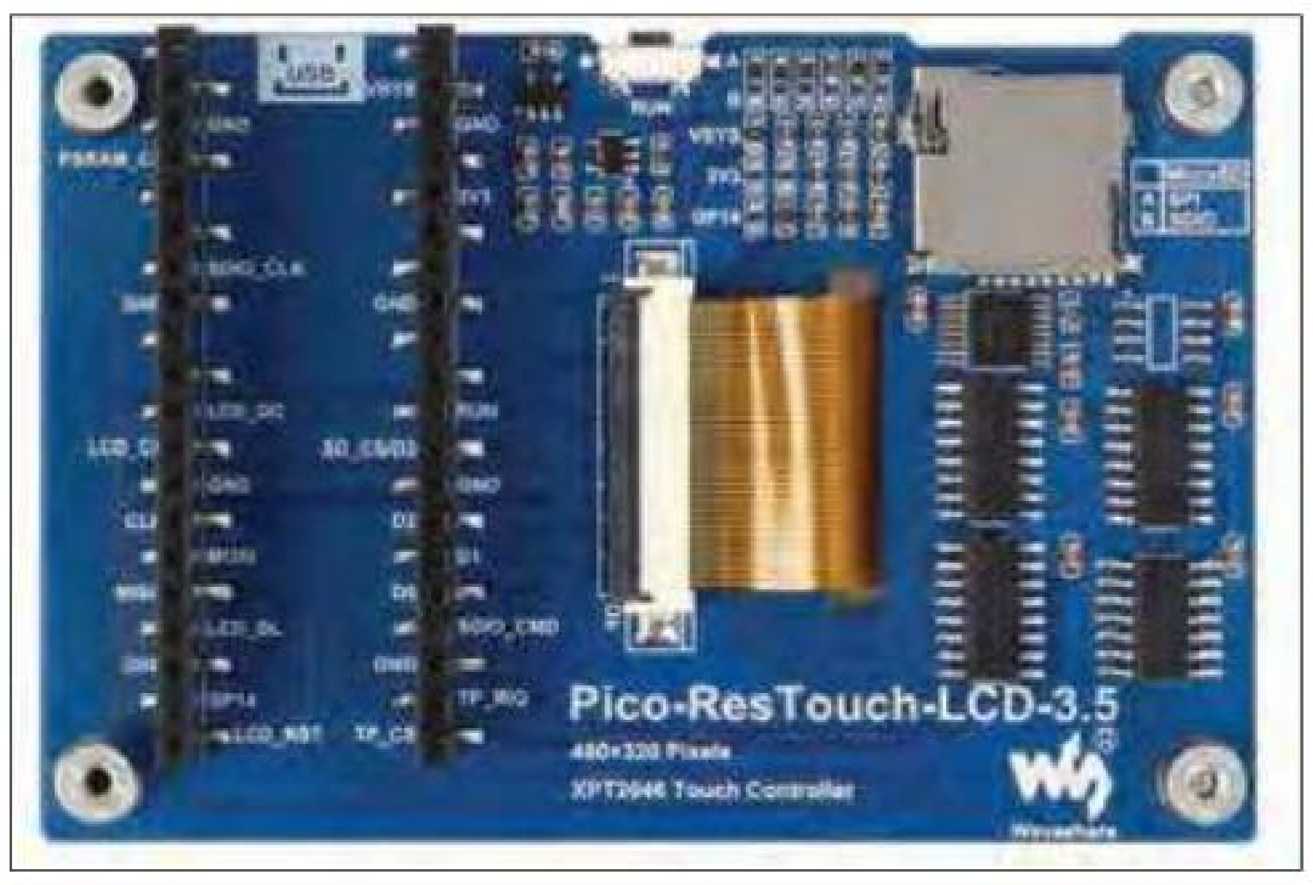


Fig.8. The Waveshare Pico-ResTouch-LCD-3.5 display (top left and above) is similar to the 2.8-inch model (lower left), apart from being slightly bigger and having a higher pixel resolution (480x320).

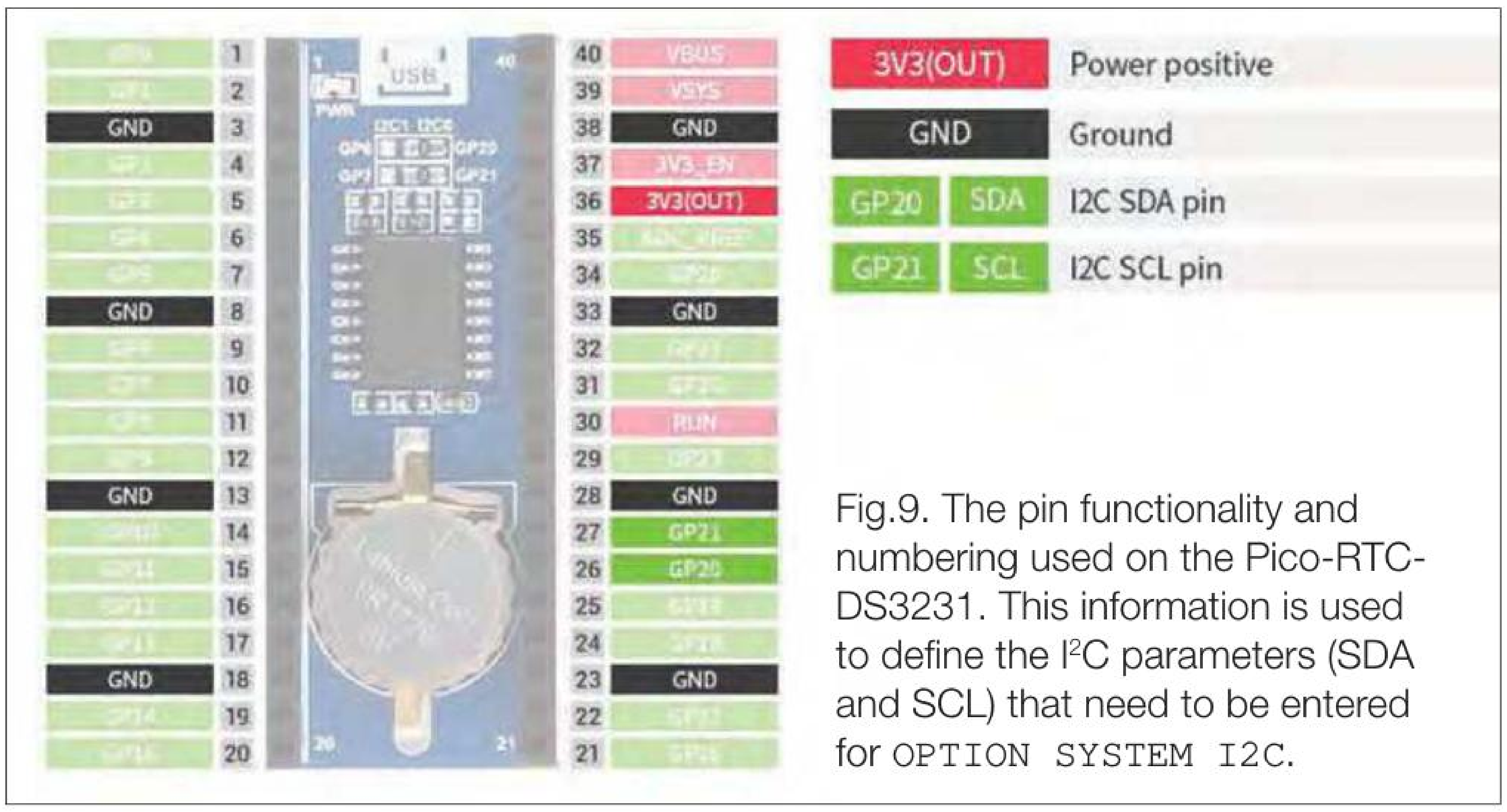


Fig.9. The pin functionality and numbering used on the Pico-RTC-DS3231. This information is used to define the I^{2}C parameters (SDA and SCL) that need to be entered for `OPTION SYSTEM I2C`.

just £4 more than the smaller 2.8-inch). Again, more details about this display module can be found at: **www.waveshare.com/wiki/Pico-ResTouch-LCD-3.5**

Rather than discuss the process step-by-step, we will instead simply summarise the 12 steps required (which are essentially identical to the steps performed above for the 2.8-inch *BackPack*).

1. **`OPTION RESET`** (clear any previous `OPTION`s that were saved)
2. **`OPTION LIST`** (check all `OPTION`s have been cleared)
3. **`OPTION CPUSPEED 252000`** (run PicoMite at 252MHz)
4. **`OPTION SYSTEM SPI GP10, GP11, GP12`** (configure SPI bus)
5. **`OPTION LCDPANEL ILI9488W, RL, GP8, GP15, GP9, GP13`** (configure the specific display module – note: use reverse-landscape orientation with this display module)
6. **`GUI TEST LCDPANEL`** (check screen set up correctly)
7. **`OPTION TOUCH GP16, GP17`** (configure touch)
8. **`GUI CALIBRATE`** (calibrate touch)
9. **`GUI TEST TOUCH`** (check touch calibration)
10. **`OPTION SDCARD GP22, GP5, GP18, GP19`** (configure SD socket – note: we need to define the SPI pins with this display module)
11. **`FILES`** (check SD card)
12. **`OPTION LIST`** (check that all `OPTION`s parameter values are correct).

Once you have successfully followed these twelve steps, you will have a working 3.5-inch *PicoMite BackPack*.

Steps for using other displays

From the above two examples describing the steps required to set up a *BackPack*, you will see that there are only four steps (4, 5, 7, and 10) that need changes to parameter values, depending on the specific display module being used. Refer to the display module's datasheet for the relevant pin numbers, and the type of display controller used (ie, the equivalent of Fig.5). Once you have done this a couple of times, you will find that to set up a brand new *PicoMite BackPack* will take no more than five minutes (and that includes loading the firmware onto a 'new' (ie, blank) Pico). Yes, it really is that quick to set up a powerful PicoMite module with a choice of display size that matches your project's requirements.

Adding an RTC

We will now show how easy it is to add an RTC to the *PicoMite BackPack*. For this, we will use yet another module from Waveshare; the Pico-RTC-DS3231 (see Fig.2; and for further details, visit: **www.waveshare.com/wiki/Pico-RTC-DS3231**). One thing to point out here is the cost – it is typically around £9.50 which by itself isn't too bad, but when compared to the cost of the Pico module, the RTC costs almost three times as much! Anyway, let's proceed on the basis that we can indeed afford this particular RTC module (although do see the next section if you have an alternative DS3231 module available).

The pinout of the Waveshare RTC module is shown in Fig.9; again, it will be a useful reference to help understand the parameter values that we use in the single `OPTION` that we need to set.

Begin by physically connecting the RTC; this can be done by either sandwiching it between the PicoMite and the display module (Fig.10a), or by inserting it into an Expander (Fig.10b).

There are just four simple steps to complete the configuration of the RTC (including testing that it works correctly):

1. First, we need to define the pins used for the I^{2}C bus (the bus used by the DS3231 RTC module). To do this, use the command: `OPTION SYSTEM I2C` *`SDA_pin, SCL_pin`* Referring to Fig.9, you will see that the command and parameters need to be entered as: **`OPTION SYSTEM I2C GP20, GP21`**
2. Next, we need to set the Date and Time in the RTC module with the command: `RTC SETTIME` *`year, month, day, hours, minutes, seconds`*

   If the date is 3 July 2022, and the time is 4:45 pm, then the command would be entered as: **`RTC SETTIME 22,7,3,16,45,00`** (check with an Internet-based clock: **https://time.is**). When you press `Enter` for the above command, the precise date and time values will be sent to the RTC module; and the RTC will manage them until either the **`RTC SETTIME`** command is used again, or the RTC's onboard battery is removed (or expires – although it should last many years before it needs replacing). Note that if you see an error message along the lines of: 'RTC not responding', then this means one of two things: either

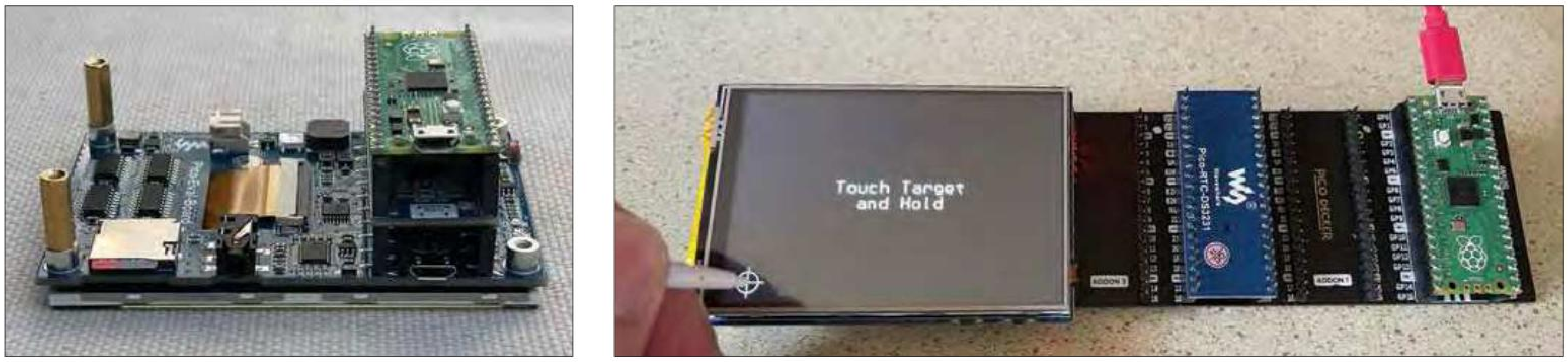


Fig.10. a) (left) The Waveshare Pico-RTC-DS3231 sandwiched between the PicoMite and the display module; b) (right) The Waveshare Pico-RTC-DS3231 inserted into a quad Expander module.

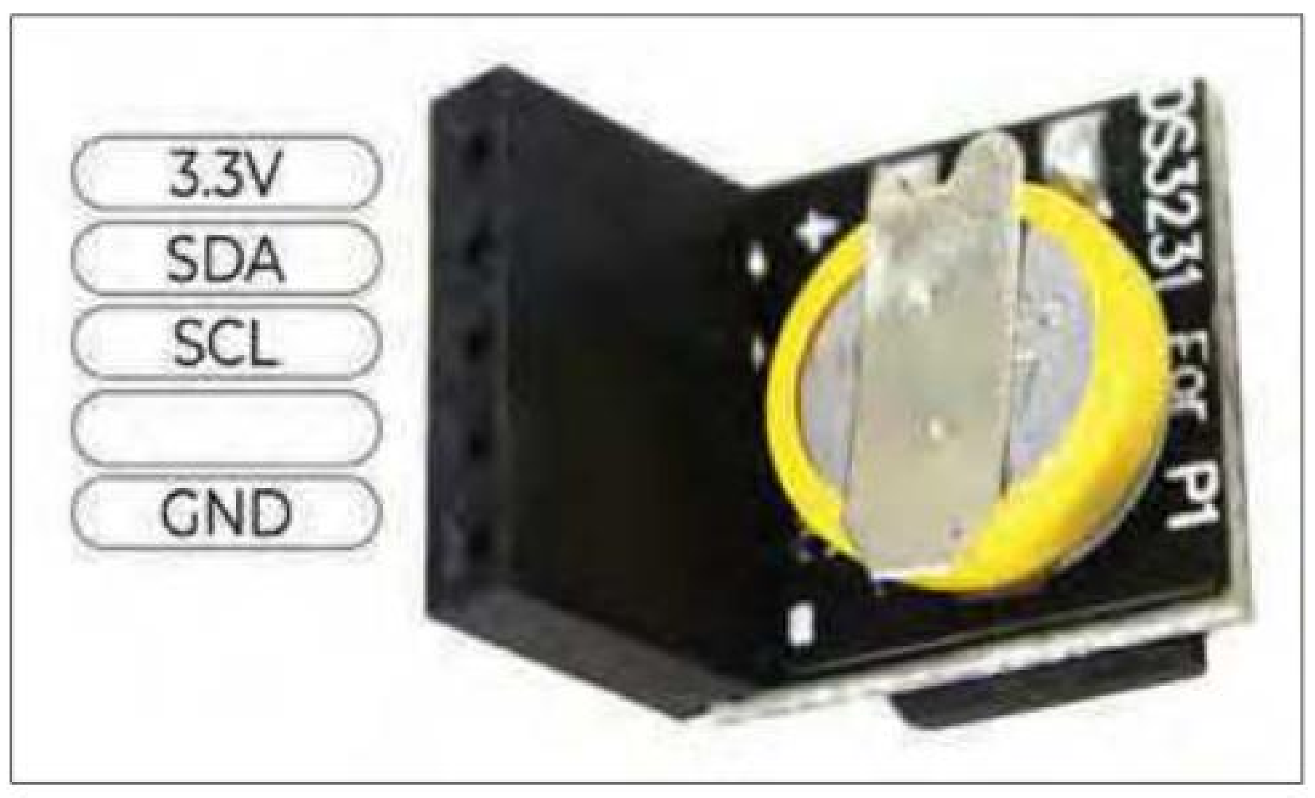


Fig.11. This is a low-cost alternative RTC module. The four labelled pins need to be connected to the PicoMite (see text).

the RTC is not inserted correctly (check it's not rotated 180°!), or the I^{2}C parameters are incorrect (check by using `OPTION LIST` and make sure that `GP20` and `GP21` have been correctly entered).

3. Once the RTC's date and time have been set up, there is one more `OPTION` that we advise setting: **`OPTION RTC AUTO ENABLE`** – it will force the firmware to upload the correct date and time from the RTC into the two MMBASIC system strings: `DATE$` and `TIME$`. This 'upload' is performed automatically whenever the PicoMite is initially powered up, and thereafter every hour.
4. To test the RTC has been set up correctly, simply remove power from the PicoMite, then reconnect power (the easiest way to do this is to remove one end of the USB lead that connects the PicoMite to your computer, then reinsert it). Then, at the command prompt, type: **`PRINT DATE$`** followed by **`PRINT TIME$`**. On pressing the `Enter` key, you should see the correct date and time returned. If not, check the RTC's battery! Note that if the RTC is disconnected from the PicoMite (when `OPTION RTC AUTO ENABLE` is set), then the attempted 'upload' will result in an error message ('the RTC cannot be seen') and this `OPTION` will also be automatically disabled. In addition, if this situation arises at the time of an hourly 'update' and while a program is running, the program will stop, and the error message displayed.

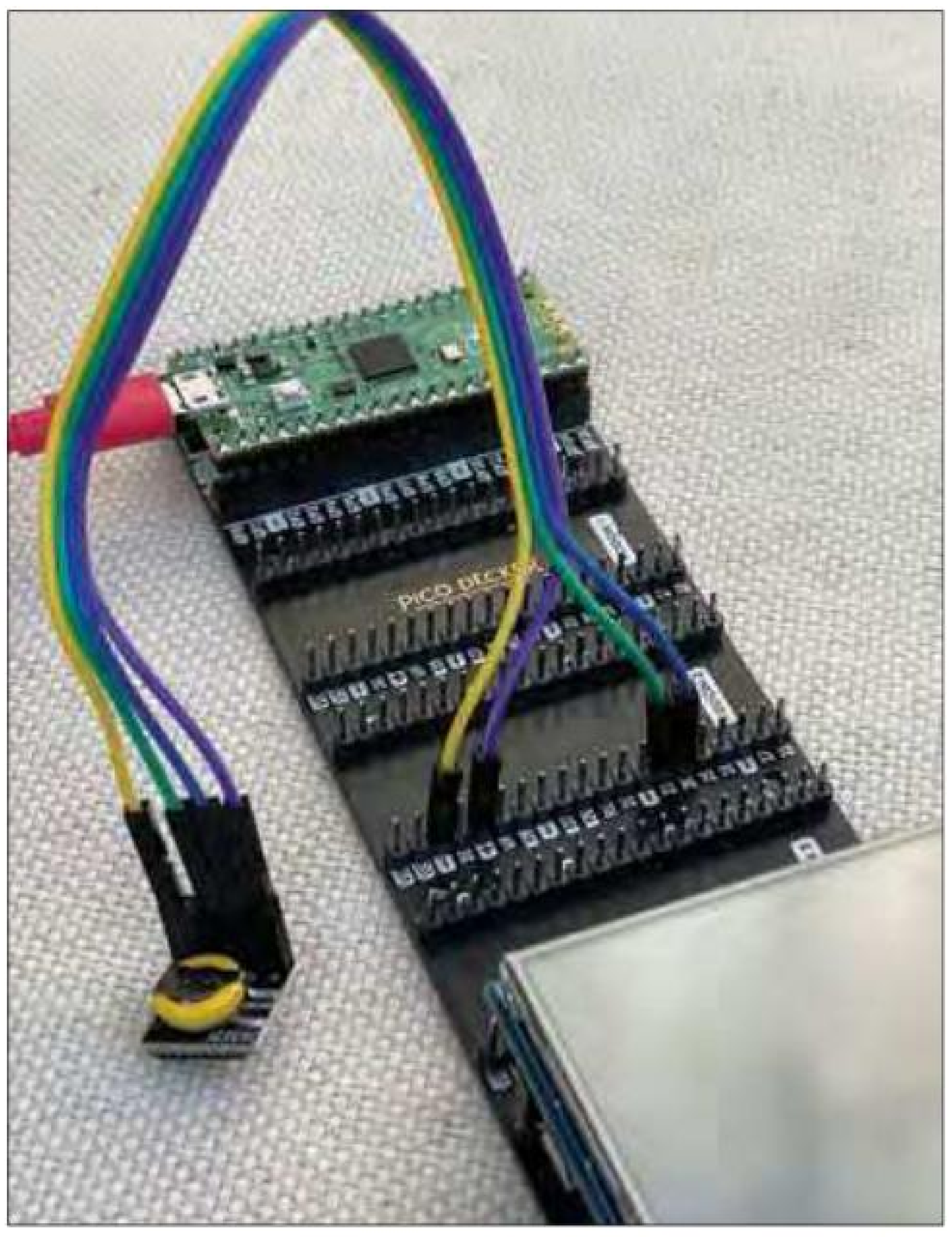


Fig.12. The alternative lower-cost RTC cannot be plugged directly into the PicoMite, so an Expander module and four Dupont leads are used to connect it.

After following these four steps, you'll have a fully functioning *PicoMite BackPack* complete with an RTC. However, as mentioned above, you may already have a different type of RTC module, so let's see what alternative steps you need to set one up.

Using a non-Pico RTC

Let's use the cheaper (£2) RPi mini DS3231 module (see Fig.11). There are just four connections we need to make; two for power (3.3V), and two for I^{2}C communication (SDA and SCL). This RTC module has a 5-way socket, so we'll use four female-male Dupont leads.

The datasheet for the DS3231 shows that the RTC can run at either 5V or 3.3V; however, it is important that we use 3.3V when connecting it to the PicoMite. As mentioned last month, the maximum input voltage to any PicoMite pin is 3.6V, and if we accidently power the RTC with 5V, then the I^{2}C pins (being pulled high by default) will also be at 5V. To eliminate any risk to the Pico module, ensure you connect the RTC power pin (labelled '+') to the Pico's 3.3V output pin. In last month's Fig.7, the 3.3V pin on the PicoMite is the fifth pin down from the top-right corner (pin 36). For the GND connection, there are several 0V pins to choose from; for example, the third pin down from the top-right corner (pin 38).

To make the two I^{2}C connections we will use the same pins on the PicoMite as used in the example above for the RTC Pico module. Hence GP20 is connected to the RTC's SDA pin (labelled 'D'), and GP21 to the RTC's SCL pin (labelled 'C').

The four connections between the RTC module and an expander module are shown in Fig.12. Note that the pins are 'flipped' on the pin-headers so that the left-hand row (in a set of header pins) is essentially the right-hand row on the PicoMite; simply follow the silkscreen printed legends on the expander module – they are clearly marked (this is exactly why we find an Expander module so useful for tinkering). In Fig.12, the coloured leads are as follows: yellow = 0V, purple = 3.3V, blue = SDA, and green = SCL.

Now that we have connected the RTC to the PicoMite, we can configure the PicoMite to use it. We have used the same I^{2}C pins as in the previous example, so you can follow the same four steps to configure, set, and test the RTC module. So, now perform the four steps to ensure everything works as expected.

RTC demo software

We have made a short demo program (**RTCdemo.txt**) available for download from the August 2022 page of the *PE* website. It displays the day, date and time on your *BackPack's* screen (see Fig.13). Note that as supplied, it can be used on any screen that has a horizontal pixel-resolution of at least 320 pixels; hence it is ideal for either the 2.8-inch or 3.5-inch *BackPack* described in this article. That said, it is very easy to follow what is happening in the code, and so it can be modified to run properly on different sized (ie, smaller) screens.

Do test it on your new RTC *PicoMite BackPack*, it demonstrates how, with just three low-cost modules and a few lines of MMBASIC code, you can quickly create a working solution; in this case an accurate clock – you can never have too many clocks!

Summary

This month, we have given you the tools to create a *PicoMite BackPack*. All you need is the PicoMite created last month, to which you add a compatible display of your choice, set up with the relevant `OPTION`s. In addition, you should also understand how to use other readily available plug-in Pico Modules with a PicoMite – it is just a

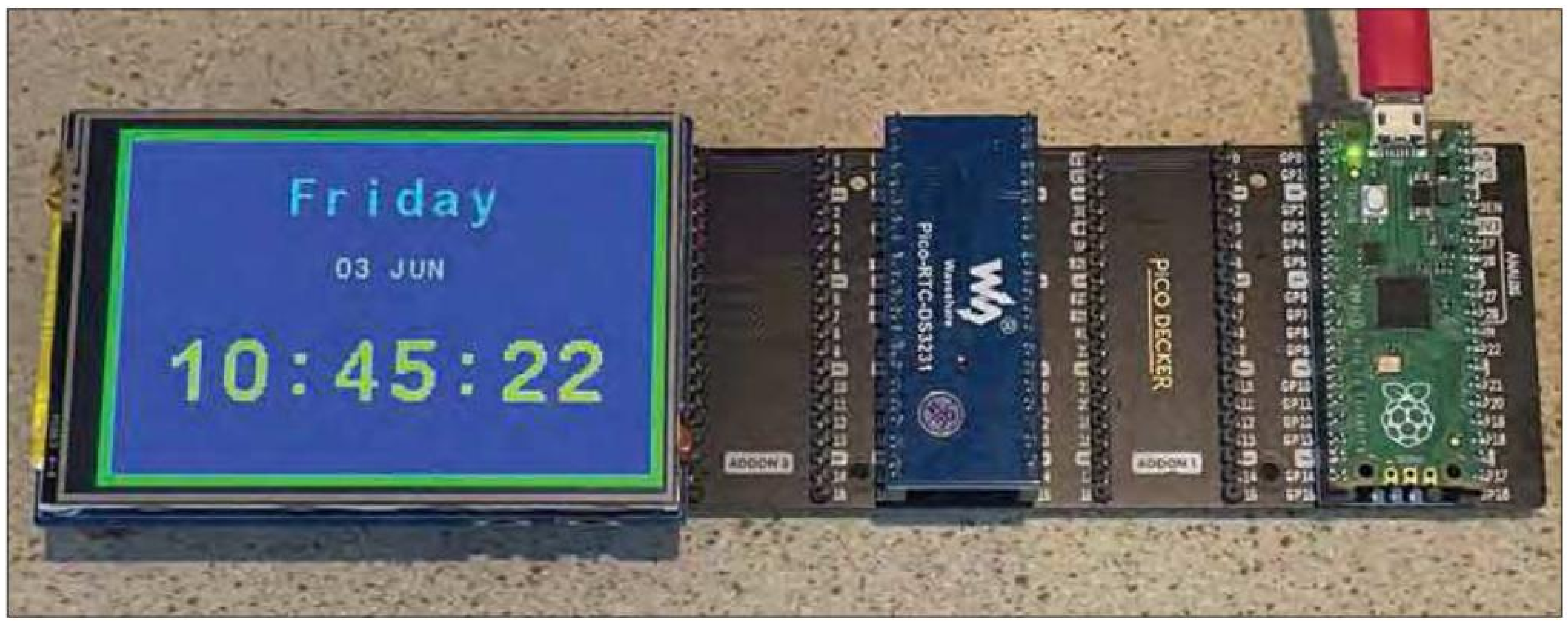


Fig.13. The **RTCdemo.txt** program in action (running on a PicoMite that has been assembled on a quad Expander module).

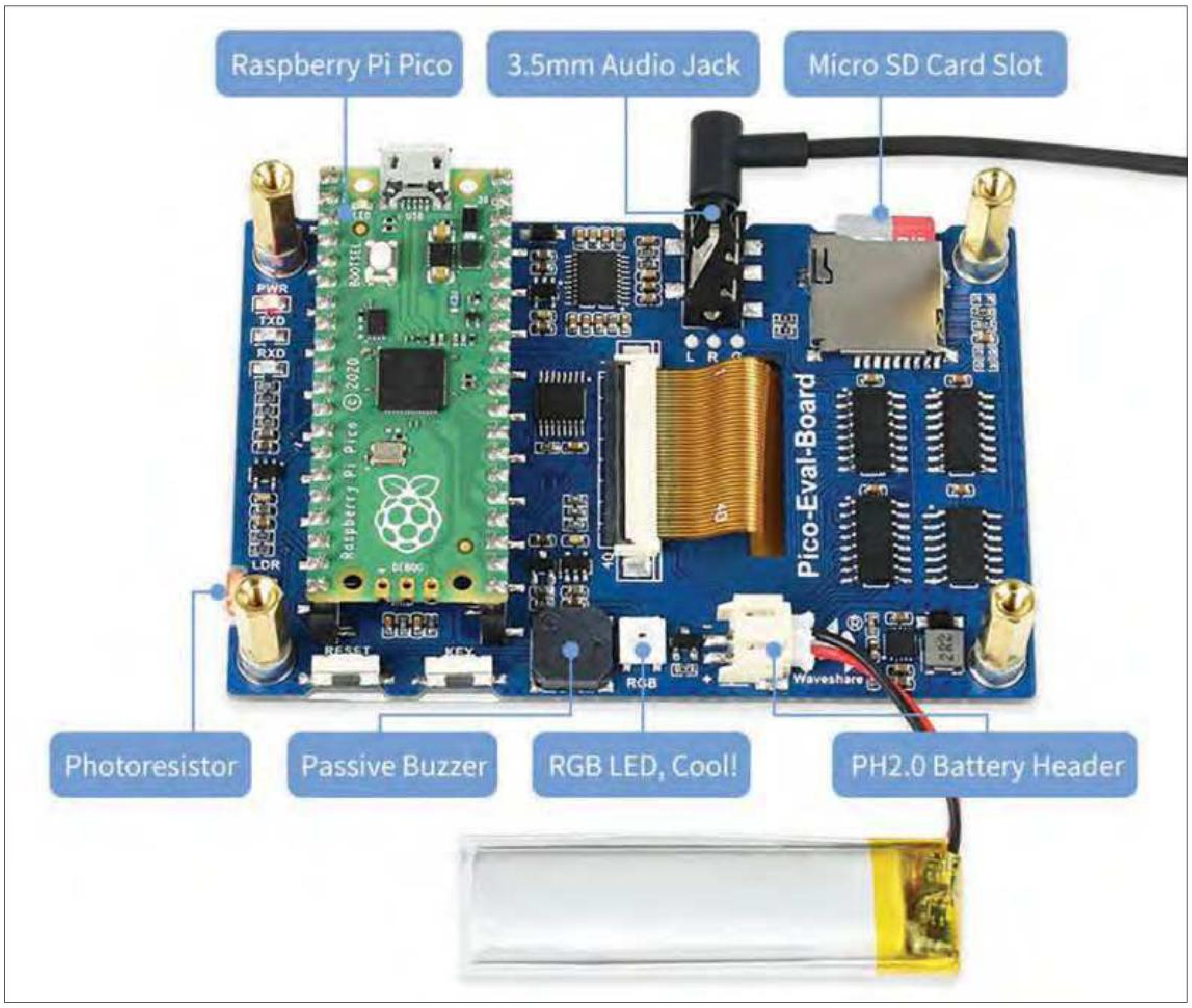


Fig.14. As well as having a 3.5-inch IPS touchscreen and a micro-SD socket onboard, the Pico-Eval-Board has additional components. It is the perfect display for building an experimental *PicoMite BackPack*.

matter of understanding the pin-numbers used on the module being attached, and setting any `OPTION`s accordingly. Furthermore, you are not limited to using plug-in Pico modules as we saw when we hooked up a mini RTC module; again, it is just a matter of understanding and identifying the connections available on the module being used.

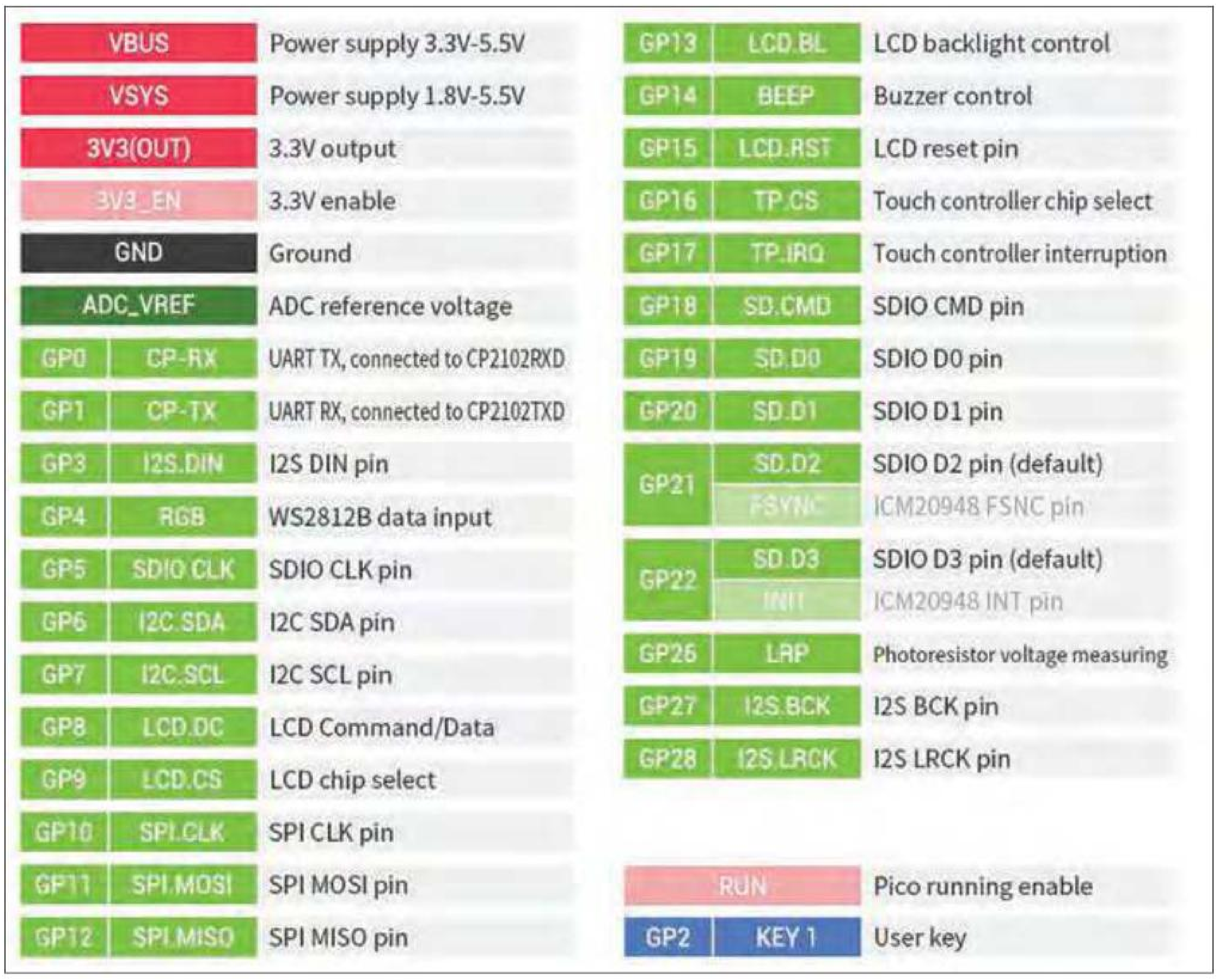
It's worth searching online to see what other Pico modules exist, and if any of them could assist you in quickly creating your next project idea. By using modules (and Dupont leads), it allows you to quickly do 'proof-of-concept' tests. Note too, it is possible to build a complete project without soldering (not what you would expect to read in an electronics magazine), however, the main point we're stressing here is that the *PicoMite BackPack* is a powerful building block that you can use in virtually any project – whether that project requires a soldering iron, or not.

Pico-Eval-Board

The 3.5-inch screen used on the 3.5-inch display module discussed above is also used on the Pico-Eval-Board (see Fig.14). This Pico module contains many useful components, such as a buzzer, a mini-button, a photoresistor for measuring light level, an RGB LED (W2812B), a 9-axis motion sensor (ICM20948), a LiPo battery header (complete with recharging circuit), and a reset button (which, is also present on the two display-modules discussed earlier). For more details, visit: **www.waveshare.com/pico-eval-board**

Fig.15 shows the pin numbers used to connect to the various onboard components. It is shown here to inspire anyone opting to use this module in their *BackPack* to access some of these other components. For example, take a look in the *User Manual* at the `BITBANG WS2812` command to see how easy it is to control the RGB LED. And for anyone up for a challenge, try reading data from the 9-axis motion sensor!

The cost of this module is typically around £29.50, which at just £11 more than the 3.5-inch display module, represents good value for money. (It is the actual module used in the photos in this article showing the 3.5-inch *PicoMite BackPack*.)



VBUS		Power supply 3.3V-5.5V
VSYS		Power supply 1.8V-5.5V
3V3(OUT)		3.3V output
3V3_EN		3.3V enable
GND		Ground
ADC_VREF		ADC reference voltage
GP0	CP-RX	UART TX, connected to CP2102RXD
GP1	CP-TX	UART RX, connected to CP2102TXD
GP3	I2S.DIN	I2S DIN pin
GP4	RGB	WS2812B data input
GP5	SDIO CLK	SDIO CLK pin
GP6	I2C.SDA	I2C SDA pin
GP7	I2C.SCL	I2C SCL pin
GP8	LCD.DC	LCD Command/Data
GP9	LCD.CS	LCD chip select
GP10	SPI.CLK	SPI CLK pin
GP11	SPI.MOSI	SPI MOSI pin
GP12	SPI.MISO	SPI MISO pin
GP13	LCD.BL	LCD backlight control
GP14	BEEP	Buzzer control
GP15	LCD.RST	LCD reset pin
GP16	TP.CS	Touch controller chip select
GP17	TP.IRQ	Touch controller interruption
GP18	SD.CMD	SDIO CMD pin
GP19	SD.D0	SDIO D0 pin
GP20	SD.D1	SDIO D1 pin
GP21	SD.D2	SDIO D2 pin (default)
	FSYNC	ICM20948 FSNC pin
GP22	SD.D3	SDIO D3 pin (default)
	INT1	ICM20948 INT pin
GP26	LRP	Photoresistor voltage measuring
GP27	I2S.BCK	I2S BCK pin
GP28	I2S.LRCK	I2S LRCK pin
RUN		Pico running enable
GP2	KEY 1	User key

Fig.15. The pin functionality and numbering used on the Pico-Eval-Board. Refer to this (and *PicoMite User Manual*) to see if you can interact with some of the additional components.

Next time

At the time of writing, I am still waiting for various Pico modules that should have been delivered by now – the world's silicon supply chains are still recovering. So, at this point, the next topic depends on the postman! However, I can confirm that we will be using some more Pico modules to make something fun and interesting.

Until then, stay safe, and have FUN!

Questions? Please email Phil at:
contactus@micromite.org

KickStart

by Mike Tooley

Part 9: Exploring microcontroller digital-to-analogue conversion

Our occasional *KickStart* series aims to show readers how to use readily available low-cost components and devices to solve a wide range of common problems in the shortest possible time. Each of the examples and projects can be completed in no more than a couple of hours using ‘off-the-shelf’ parts. As well as briefly explaining the underlying principles and technology used, the series will provide you with a variety of solutions and examples, along with just enough information to be able to adapt and extend them for your own use.

This ninth instalment explores ways of adding (and improving) analogue output capability from a wide range of popular microcontrollers.

Analogue-to-digital and digital-to-analogue converters (ADC and DAC respectively) provide a means of interfacing the digital world of a microcontroller to the outside analogue world. Unfortunately, while most microcontrollers incorporate reasonable ADC hardware, the same cannot be said of their DAC counterparts. We’ve put this into context in Tables 9.1 and 9.2, comparing the on-board ADC/DAC capabilities of three of the most popular microcontrollers.

Enter PWM

As noted in Table 9.2, most low-cost microcontrollers don’t incorporate true DAC; instead, they use pulse-width modulation (PWM), where the average value of a train of rectangular pulses is used to represent an analogue voltage. This process is illustrated in Fig.9.2.

Fig.9.2(a) shows a perfect square wave (ie, a rectangular wave with a 50% duty cycle). The average value of this waveform will be exactly 50% of its maximum (peak) value. In Fig.9.2(b) a repetitive pulse having a duty cycle of 25% will have a long-term average value that’s 25% of its maximum, while in Fig.9.2(c) a pulse wave with 75% duty cycle will exhibit a long-term average value which is 75% of its maximum. This leads us to the notion that we can produce a continuous range of analogue voltages by simply varying the duty cycle of a train of rectangular pulses.

PWM is a useful technique for non-critical applications, and it can be easily implemented on devices that don’t have a built-in DAC capability. All that is required is a handful of additional components that will average the output, as shown in Fig.9.3.

In Fig.9.3, R1 and C1 form a simple RC low-pass filter, while IC1 acts as a unity-gain buffer (exhibiting a very high input impedance and capable of tolerating a relatively low resistance at the output). The PWM pulse train at the GP16 output of the Raspberry Pi Pico has an amplitude of 3.3V, hence the output at pin-6 of IC1 can be made to vary over the range 0 to +3.3V.

The analogue output voltage from Fig.9.3 can be easily set using just a few

Fig.9.1. Unlike the Raspberry Pi Pico and Arduino Uno microcontrollers, this ESP32 NodeMCU development board offers a true DAC capability. Unfortunately, its two on-board DACs only provide 8-bit resolution (so only 256 unique output voltage levels are possible).

Table 9.1 Comparison of microcontroller ADC capability

Microcontroller	ADC capability	Notes
Arduino Nano	Six 10-bit ADC channels. Input voltages can range from 0 to 5V.	The ADC channels are labelled A0 to A5 on the board.
ESP32 NodeMCU (Fig.9.1)	Up to 18 12-bit ADC channels. The measurement range is limited by the 1.1V internal voltage reference. Larger inputs can be measured by applying one of four input attenuation options.	The ADC resolution is configurable (typical values are 9, 10, 11 and 12-bits). Some boards have a restricted number of ADC pins accessible. ADC2 cannot be used concurrently with Wi-Fi.
Raspberry Pi Pico	Four 12-bit ADC channels. Input voltages can range 0 to 3.3V.	One ADC channel is dedicated to the internal temperature sensor. The three remaining ADCs are available at GPIO26, GPIO27, and GPIO28.

Table 9.2 Comparison of microcontroller DAC capability

Microcontroller	DAC capability	Notes
Arduino Nano	No true DAC hardware present, but pulse-width modulation (PWM) can be used. Maximum PWM output is 5V with 8-bit resolution.	Digital I/O pins 3,5,6,9,10 and 11 support PWM (these pins are marked with a ‘~’ symbol on the board).
ESP32 NodeMCU (Fig.9.1)	Two on-board 8-bit DACs with an output range from 0 to 3.3V.	GPIO pins 25 and 26 are available for DAC.
Raspberry Pi Pico	No true DAC hardware present, but PWM can be used. Maximum PWM output is 3.3V with 16-bit resolution.	16 PWM output channels available simultaneously; each GPIO pin can be configured as a PWM output (see text).

lines of Python code. For example, the following code will produce an analogue output voltage of 1V:

```
from machine import Pin, PWM # import from the library module
out_pin = PWM(Pin(16))       # use GP16 for PWM
out_pin.freq(100000)         # set the PWM frequency to 100kHz
out_pin.duty_u16(19859)      # set the duty cycle for 1V output
```

It's possible to condense the last three lines of code into a single instruction, as follows:

```
out_pin = PWM(Pin(16), freq=100000, duty=19859) # GP16, 100kHz, 1V
```

Note that the duty cycle is represented by an integer (n) in the range 0 to 65535, where a value of 65535 represents a maximum possible duty cycle of 100%.

The following formula can be used to determine the required value for use with the `out_pin.duty_u16(n)` function:

$n = V_{out} \times 19859$

The 3.3V maximum output voltage from the circuit in Fig.9.3 can sometimes be a limitation, but this problem can be easily overcome by incorporating some gain in the buffer stage, as shown in Fig.9.4. This circuit can produce output voltages of up to about 6V.

In the circuit of Fig.9.4 (which has a gain of 2) the duty cycle required for a given output voltage can be calculated from:

$n = V_{out} \times 9930$ or $n \approx V_{out} \times 10000$

The approximate relationship can help make life easy. For example, an output of 2.5V will result from a value of 25000, an output of 5V from a value of 50000, and so on.

As a further example, the following line of code will produce an analogue output voltage of 4.5V:

```
out_pin = PWM(Pin(16), freq=100000, duty=45000) # GP16, 100kHz, 4.5V
```

Arduino analogue output

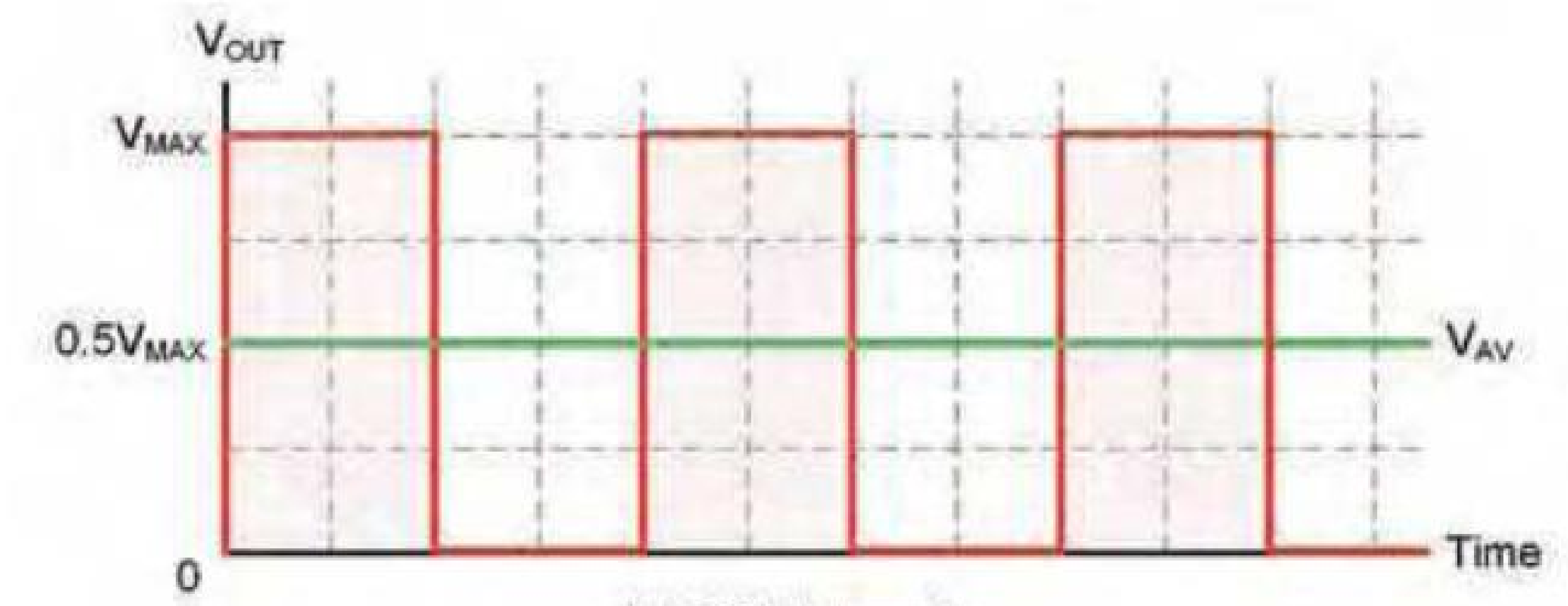
On an Arduino Uno, pins 3, 5, 6, 9, 10 and 11 are available for PWM and the somewhat misleadingly named `analogWrite()` function used to generate an analogue output voltage. At this point it is worth noting that `analogWrite()` does not actually write a value to a DAC register. Instead, it performs in the same way as the Raspberry Pi's `out_pin.duty_u16()` function, which just sets the duty cycle that will be used for PWM waveform generation. It is also worth being aware that the Uno's low pulse-repetition frequency (PRF) is approximately 490Hz on pins 3, 9, 10 and 11, increasing to around 980Hz on pins 5 and 6.

The range of values that can be used in the `analogWrite()` function is constrained by the use of 8-bit data and consequently they range from 0 to 255. Thus, a maximum analogue output corresponding to a 100% duty cycle will result from `analogWrite(255)`. Half full-scale output voltage will be produced by a square wave (50% duty cycle) using `analogWrite(128)`. With a 5V logic supply, the maximum and half-scale output voltages will amount to 5V and 2.5V respectively, with the output voltage adjustable from 0V to 5V in 256 steps, each of around 20mV.

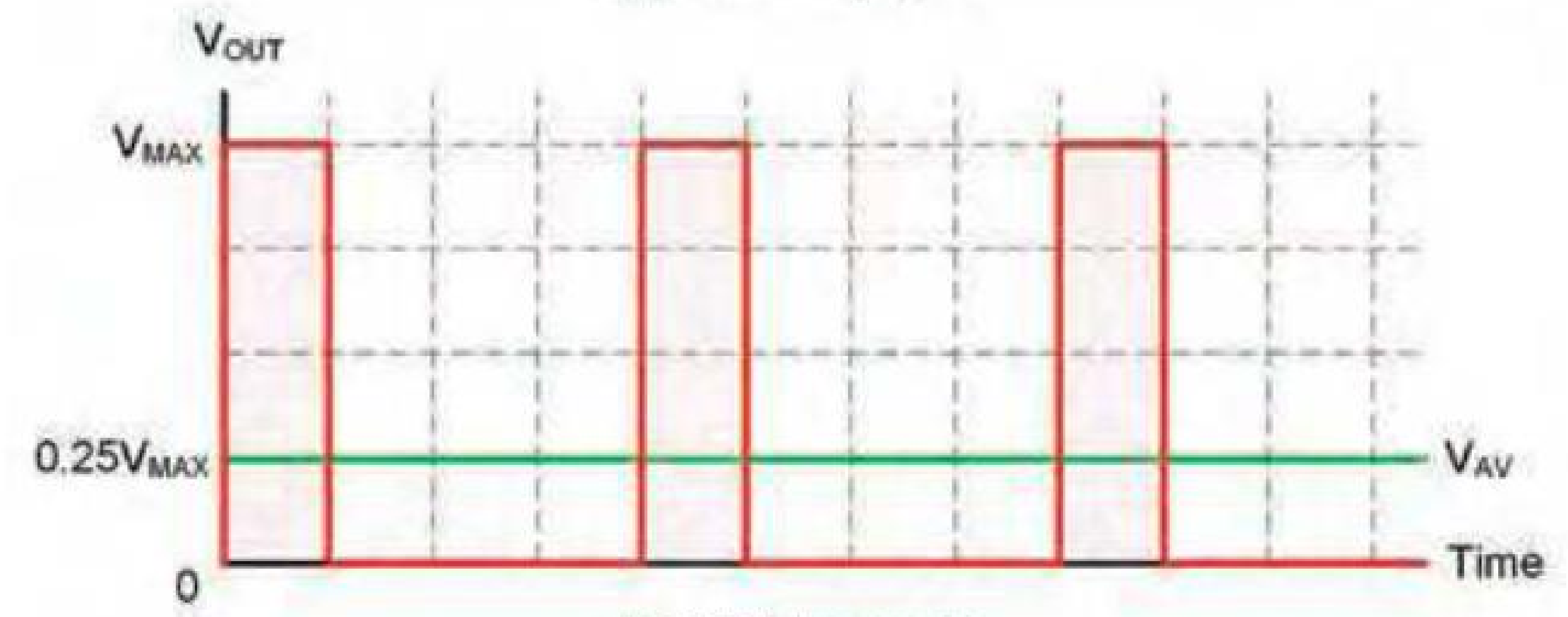
As an example, let's assume that you need to produce an analogue output voltage of 4.5V. As with the previous example, you will first need to determine the required value for use with the `analogWrite(n)` function. This can be calculated from the relationship:

$n = V_{out} \times 51$

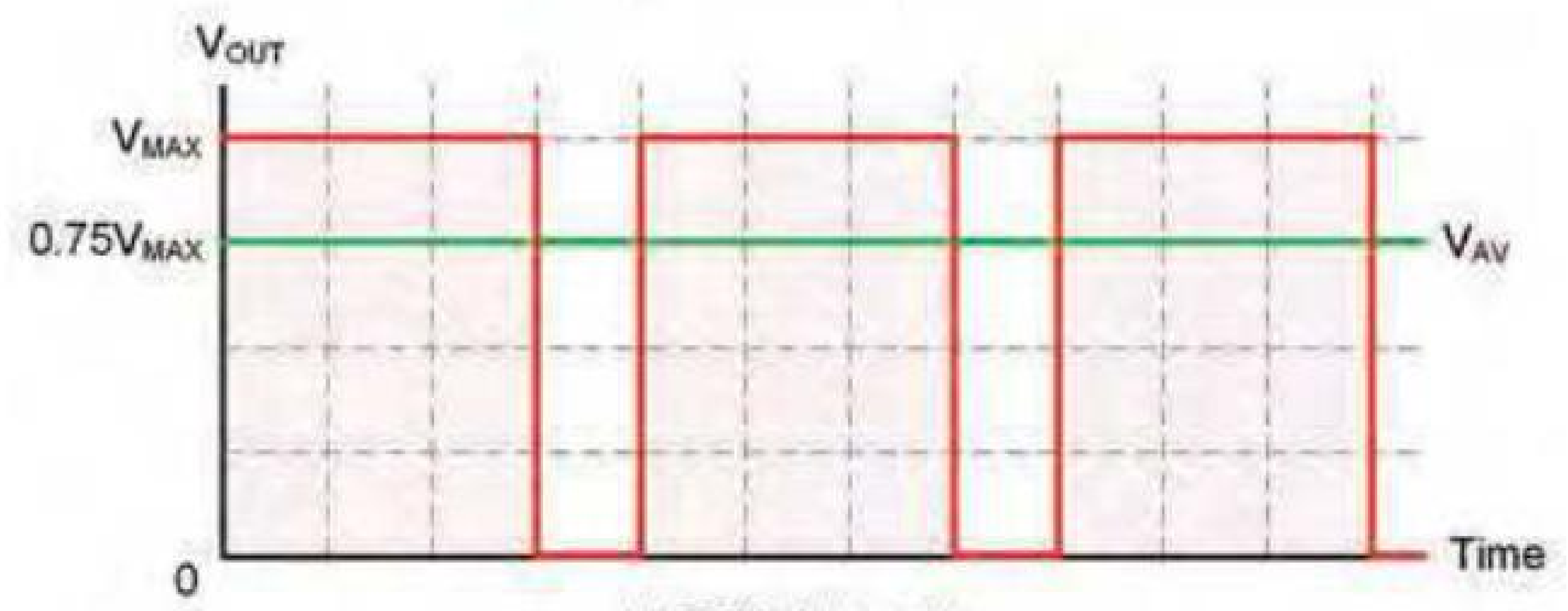
So, for 4.5V output you will need: `analogWrite(230)`



(a) 50% duty cycle



(b) 25% duty cycle



(c) 75% duty cycle

Fig.9.2. Generating an analogue voltage using PWM techniques.

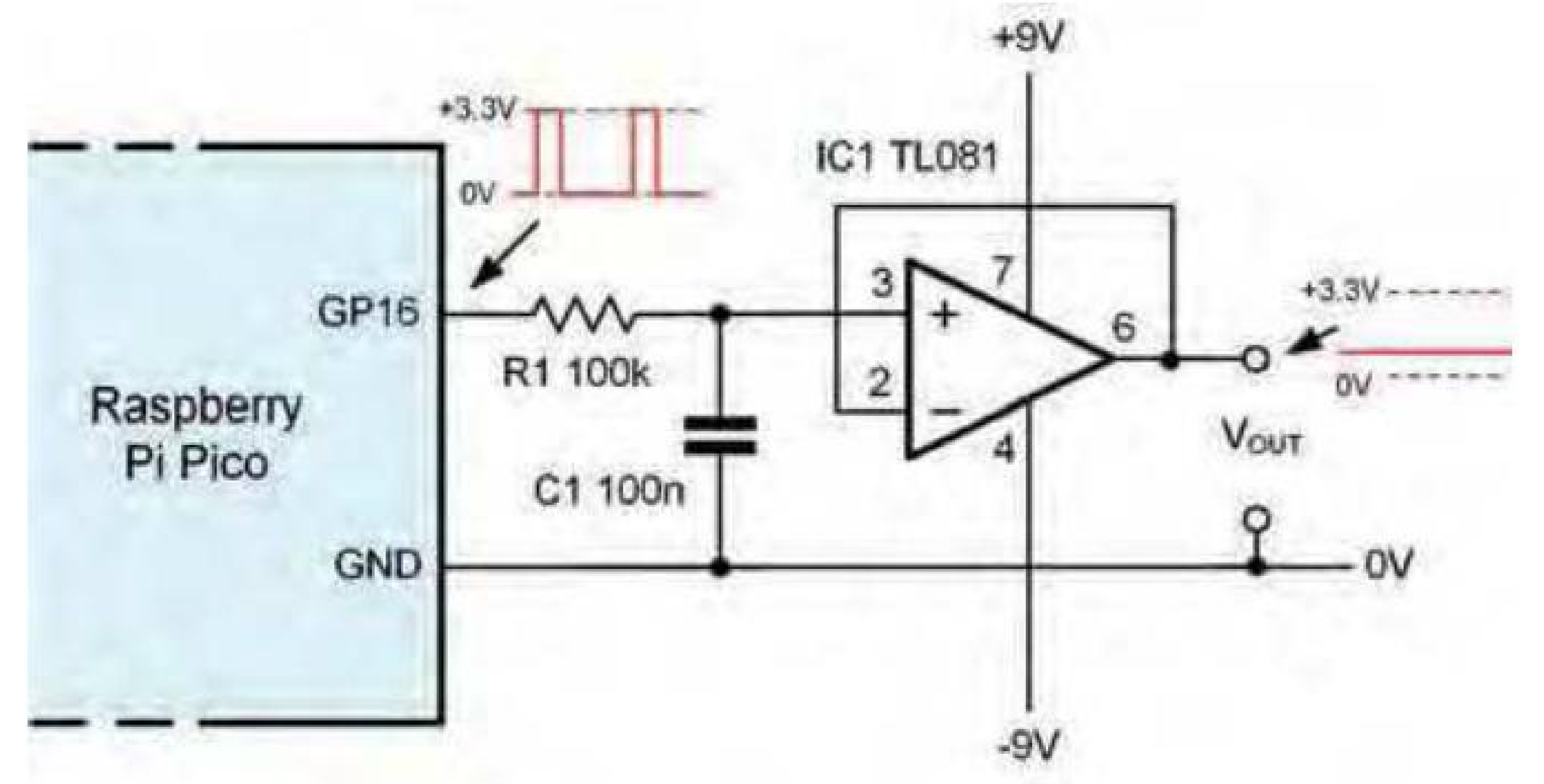


Fig.9.3. Obtaining an analogue output voltage from a Raspberry Pi Pico.

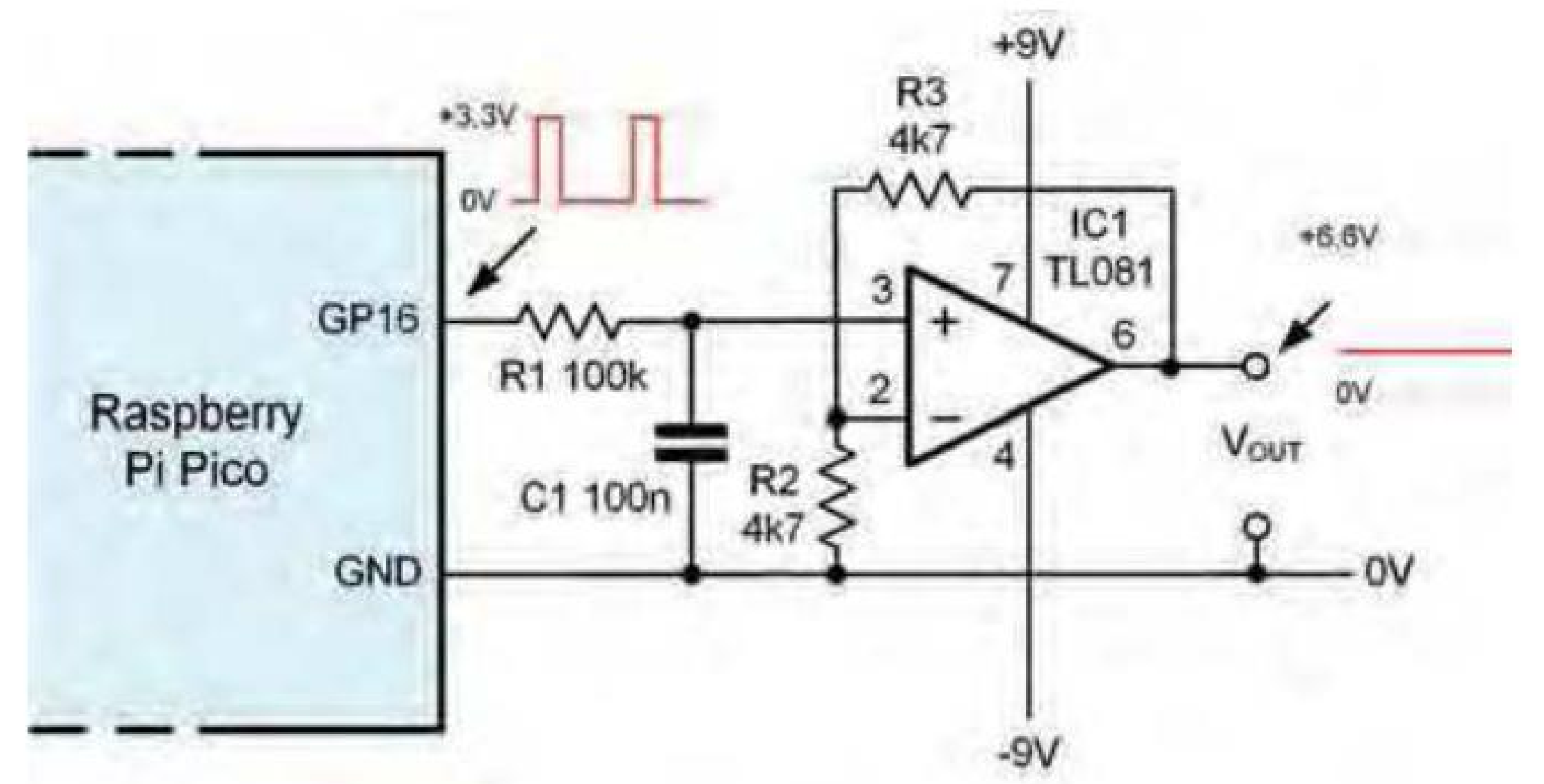


Fig.9.4. Increasing the output voltage range of Fig.9.3.

PWM can be a cost-effective solution for many simple DAC applications, but the technique has quite a few limitations, including relatively low bit resolution, slow settling time, poor power efficiency, reduced accuracy (when supply rail voltages are used as voltage references), unwanted noise generation, and the need for averaging and filtering circuitry. As a result, it often makes better sense to add a dedicated DAC module to a host microcontroller using either the I^2C or SPI bus (the latter is more suitable for high-speed applications).

Before going further, it is worth explaining some of the terminology associated with DACs, including the important relationship between resolution and number of bits used in the conversion process.

DAC resolution

The resolution of a DAC can be quoted in terms of the smallest increment of output that the DAC can produce. This small change in output voltage is that which results from a change in the least-significant bit (LSB) of the data that's written to the DAC. An alternative way of expressing DAC resolution is the number of bits used in the conversion process. As this increases, the step size in output voltage becomes smaller. The relationship is given by:

$$V_{step} = \frac{V_{ref}}{2^n}$$

Where V_{ref} is the DAC reference voltage (often 3.3V or 5V) and n is the number of bits. To put this into context, an 8-bit DAC with a 5V reference will have a step increment of:

$$V_{step} = \frac{5}{2^8} = \frac{5}{256} = 0.01953\text{V or } 19.53\text{mV}$$

As the number of bits increases the number of steps will increase and the step size decrease, as shown as shown in Table 9.3.

Table 9.3 DAC resolution

Number of bits	Number of discrete steps	Values range	Approx resolution with 5V reference
8	2^8 = 256	0 to 255	20mV
10	2^{10} = 1024	0 to 1023	5mV
12	2^{12} = 4096	0 to 4095	1.2mV
14	2^{14} = 16384	0 to 16383	0.3mV
16	2^{16} = 65536	0 to 65535	0.08mV

DAC linearity

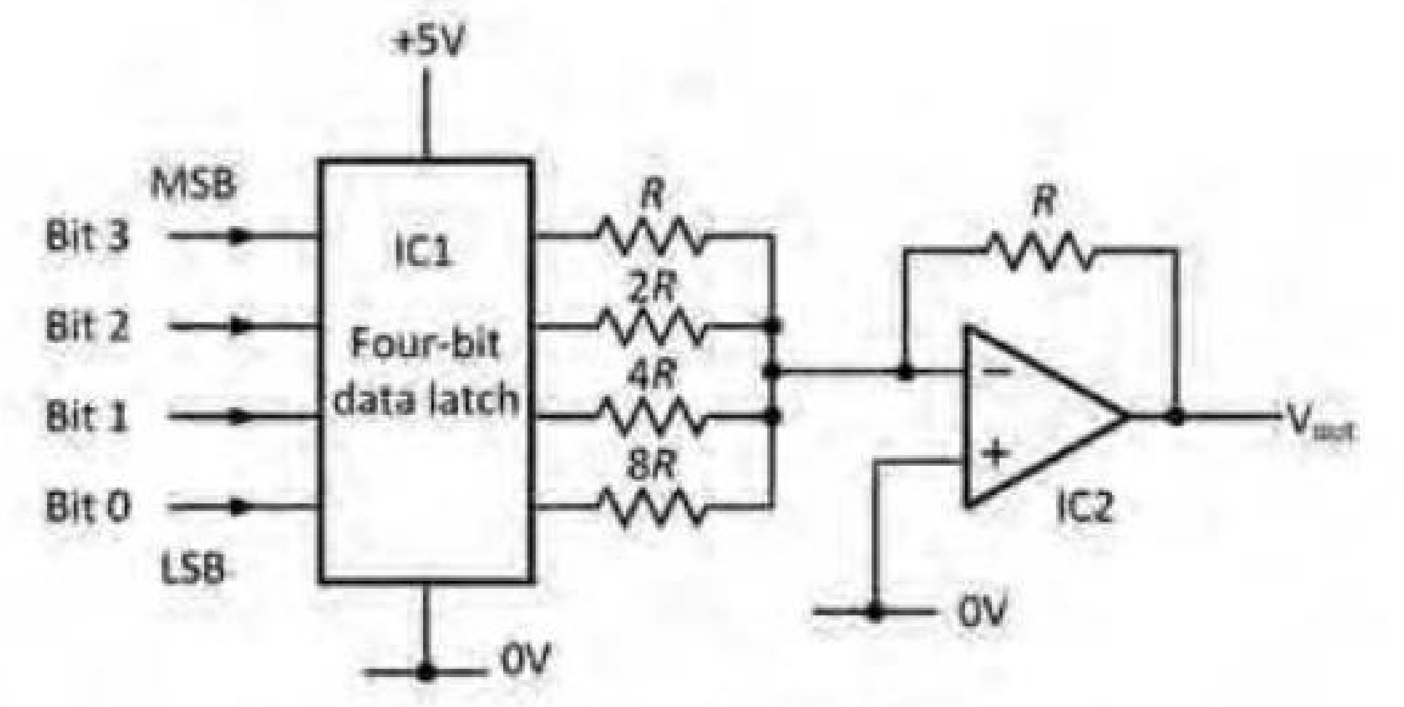
Linearity is the maximum allowable deviation from the straight line drawn between 0V and the DAC's full-scale output. Linearity can be expressed as a percentage or in terms of a fraction of the LSB. Note that the linearity of a DAC may be significantly reduced for very low (near zero) or very high (near full-scale) values.

DAC settling time

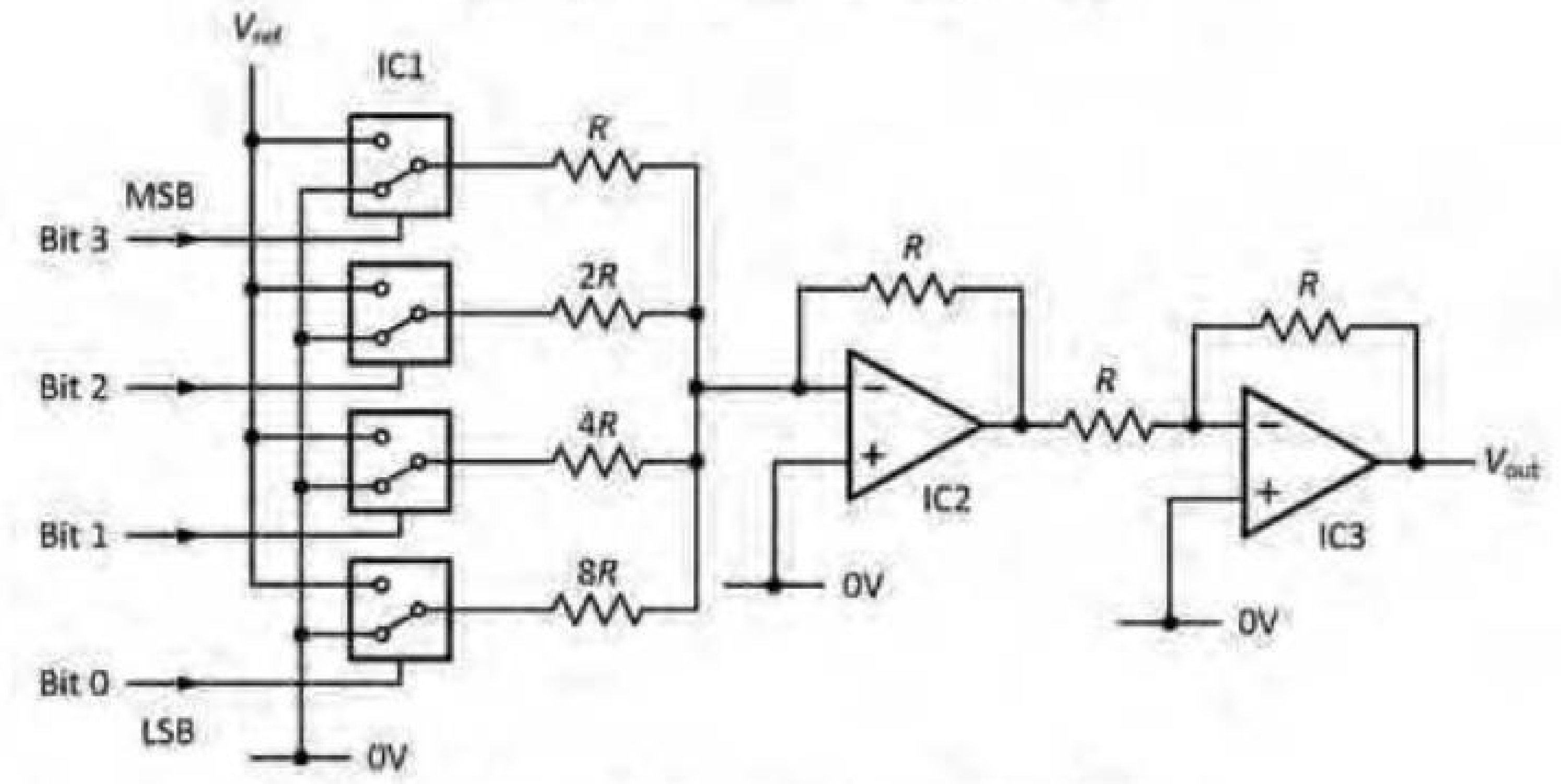
The output of a DAC cannot respond instantaneously to a change in its digital input. The delay in response is specified in terms of the time taken for the analogue output to reach its new value within specified limits. As you might expect, the worst-case settling time usually corresponds to the time taken to reach full-scale from zero.

DAC operation

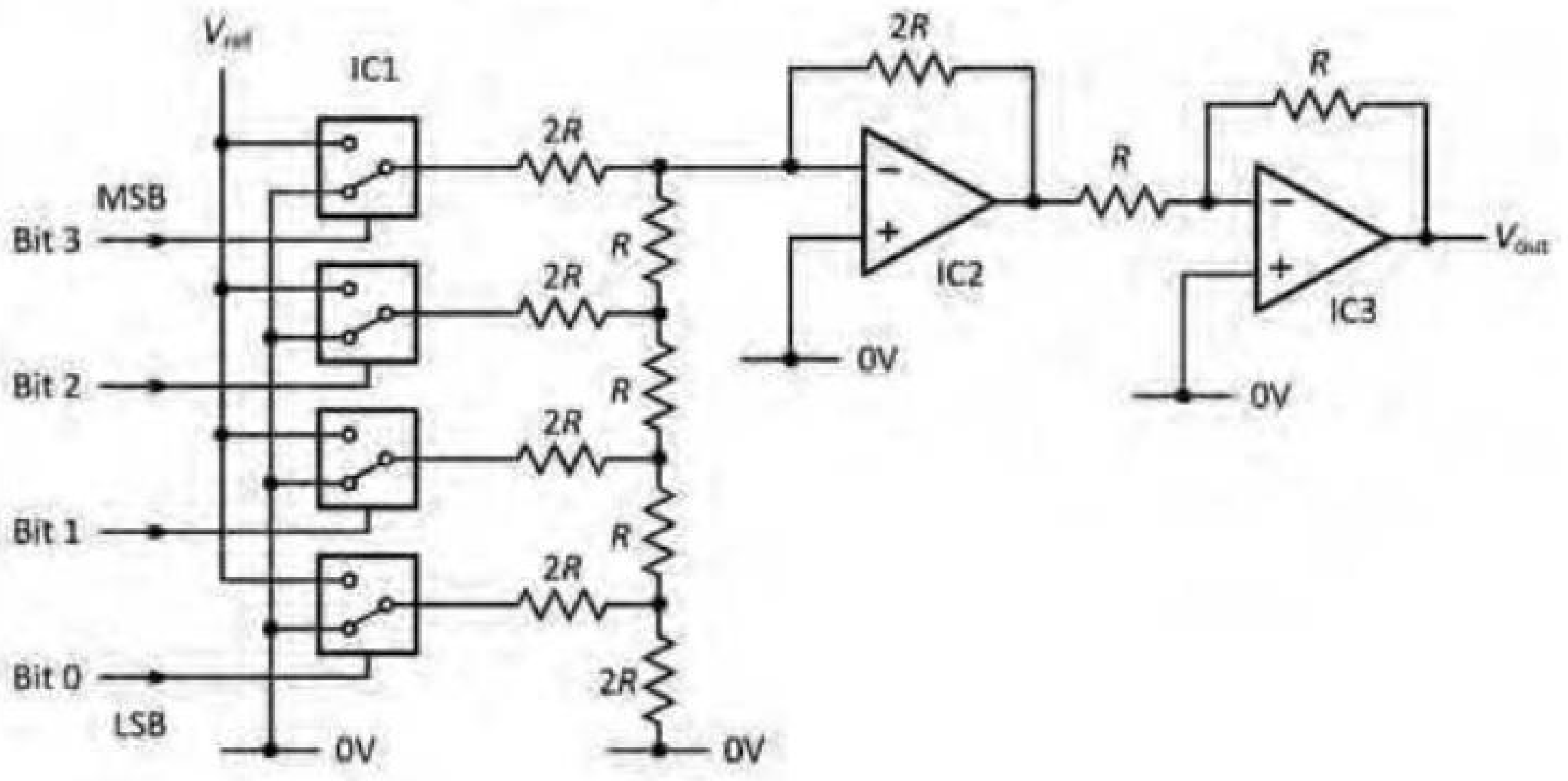
A simple form of DAC is shown in Fig.9.5(a). This uses a set of binary-weighted resistors to define the voltage gain of an operational summing amplifier (IC2). A four-bit binary latch (IC1) is used to store the binary input while it is being converted. Note that, since the amplifier is used in inverting mode, the analogue output voltage will be negative rather than positive, so a second inverting amplifier (IC3) is often connected at the output.



(a) Simple four-bit binary-weighted DAC



(b) Improved four-bit binary-weighted DAC



(c) R-2R ladder DAC

Fig.9.5. DAC ladder networks.

The voltage gains for the four inputs to IC2 are 1, 0.5, 0.25 and 0.125 for bit-3 (MSB), bit-2, bit-1 and bit-0 (LSB), respectively. If we assume that the logic levels produced by the four-bit data latch are 'ideal' (such that logic 1 corresponds to +5V and logic 0 corresponds to 0V) we can determine the output voltage corresponding to the eight possible input states by simply summing the voltages that will result from each of the four inputs taken independently. For example, when the output of the latch takes the binary value 1010 the output voltage can be calculated from:

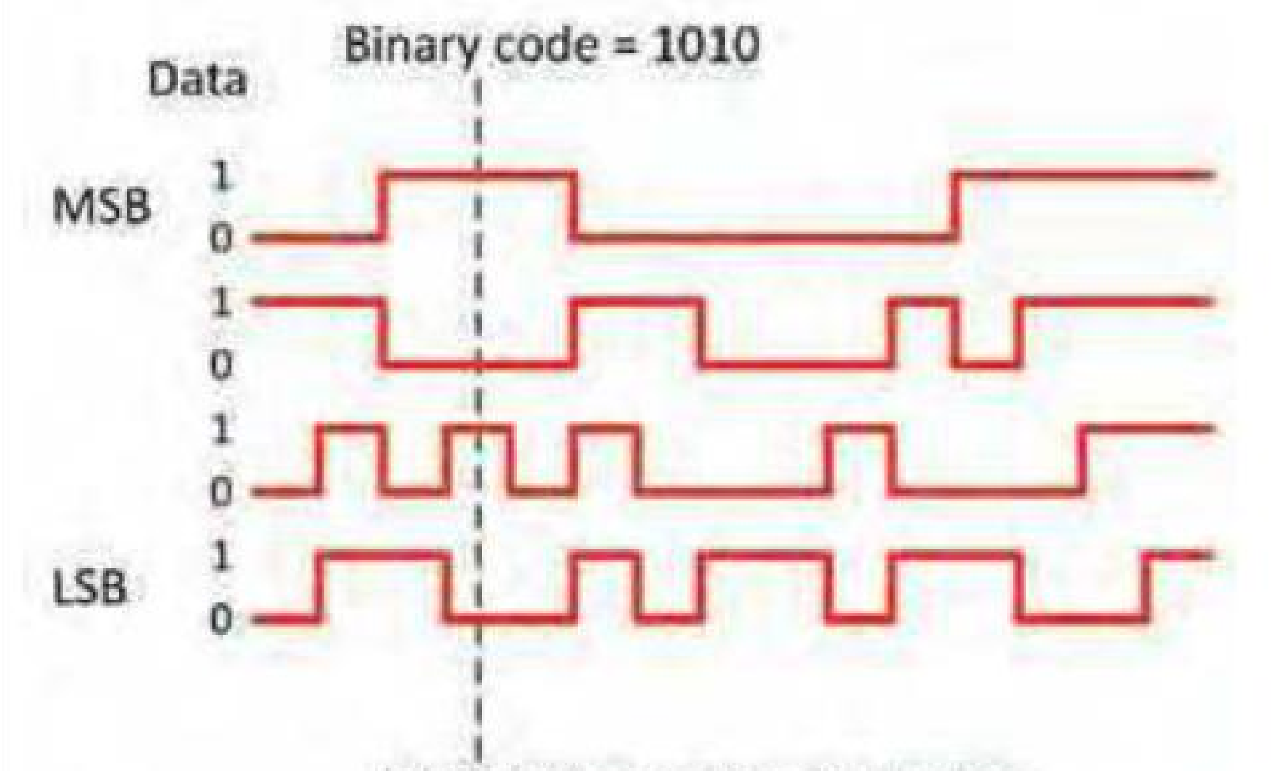
$$V_{out} = (1 \times 5) + (0.5 \times 0) + (0.25 \times 5) + (0.125 \times 0) = 6.25\text{V}$$

Similarly, when the output of the latch takes the binary value 1111 (the maximum possible) the output voltage can be determined from:

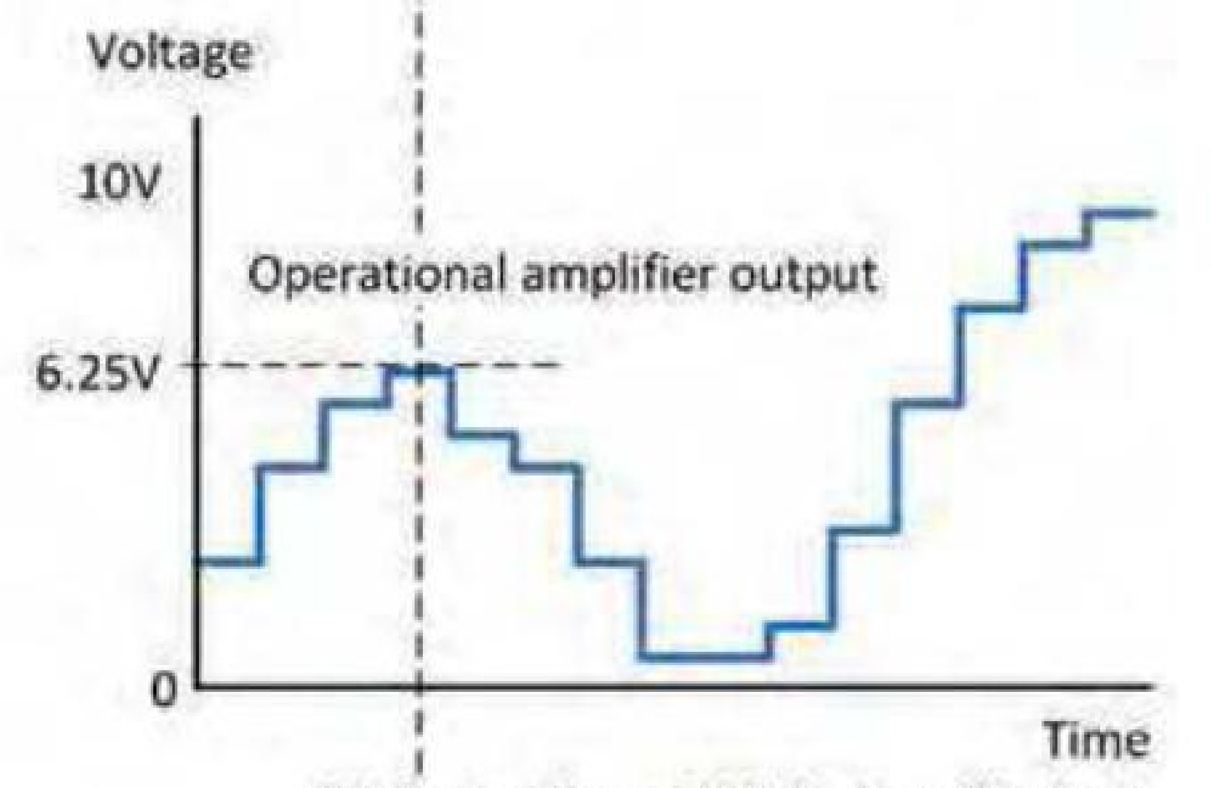
$$V_{out} = (1 \times 5) + (0.5 \times 5) + (0.25 \times 5) + (0.125 \times 5) = 9.375\text{V}$$

In this simple 4-bit DAC there are only 16 output voltage steps, ranging from 0V to 9.375V in steps of 0.625V.

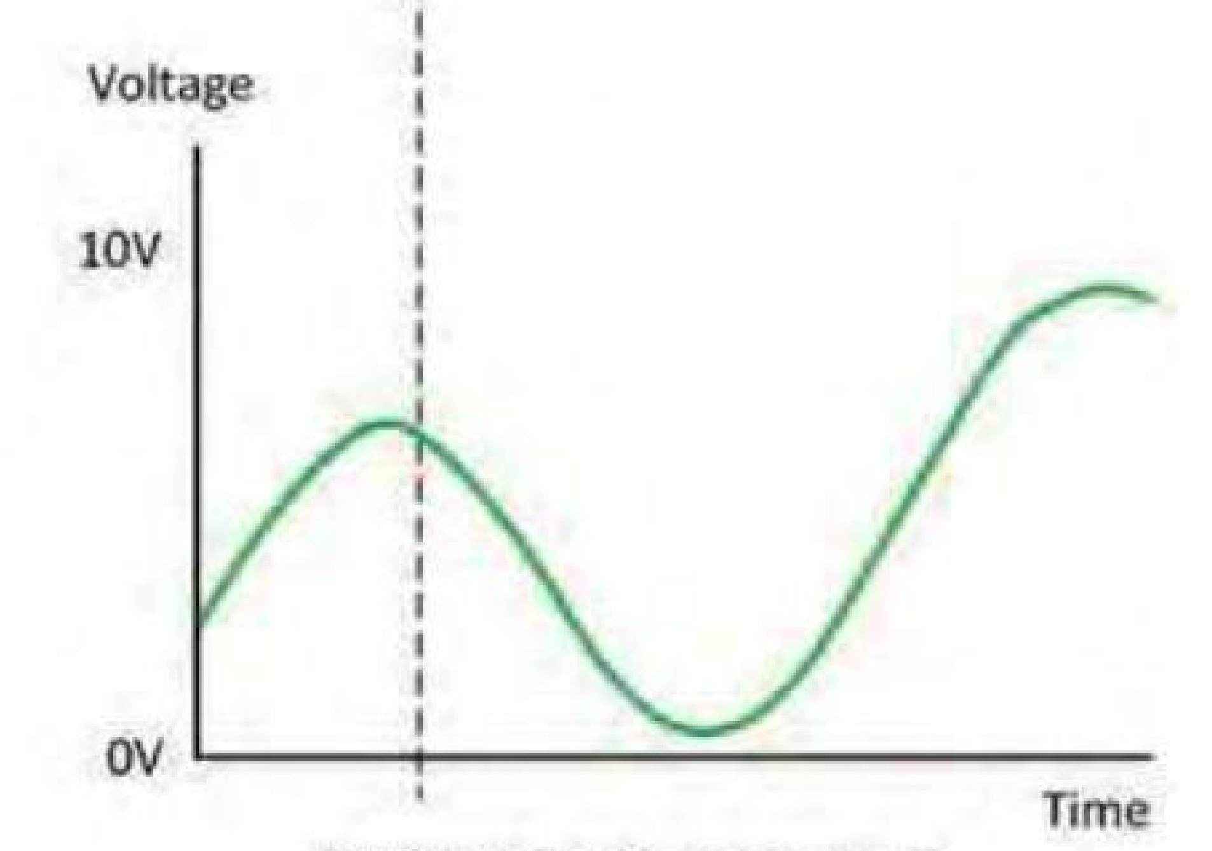
An improved binary-weighted DAC is shown in Fig.9.5(b). This circuit operates on a similar principle to that shown in Fig.9.5(a) but uses four analogue switches (IC1) instead of a four-bit data latch. The analogue switches are controlled by the logic inputs so that the respective output is connected to the reference voltage (V_{ref}) when the respective logic input is at logic 1, and to 0V when the corresponding logic input is at logic 0. When compared with the previous arrangement, this circuit offers the advantage that the reference voltage is considerably more accurate and stable than using the logic level to define the analogue output voltage.



(a) Data from microcontroller

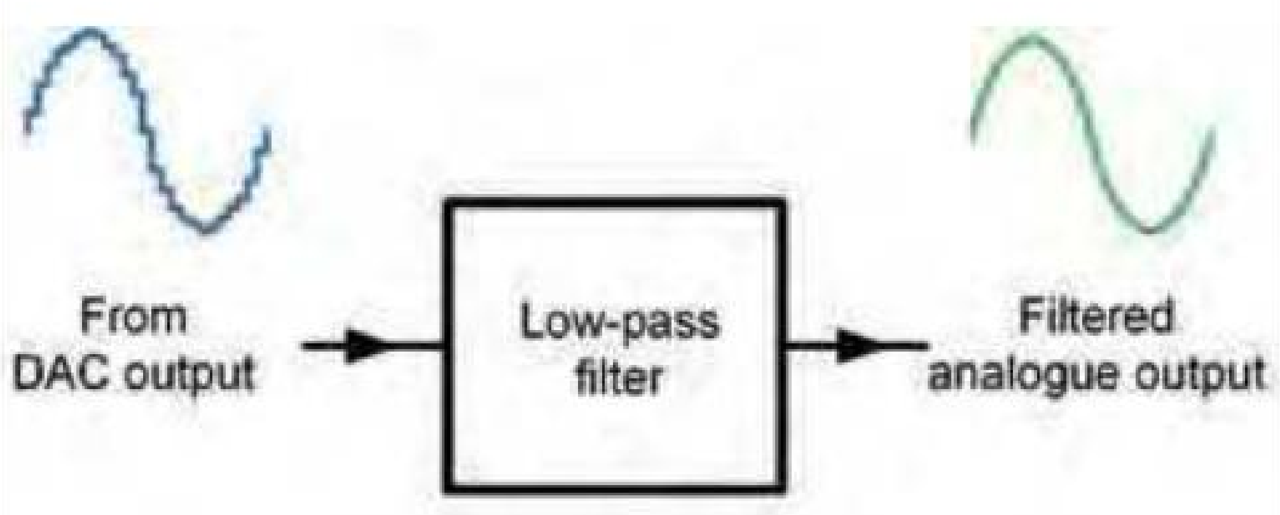


(b) Output from DAC (before filtering)

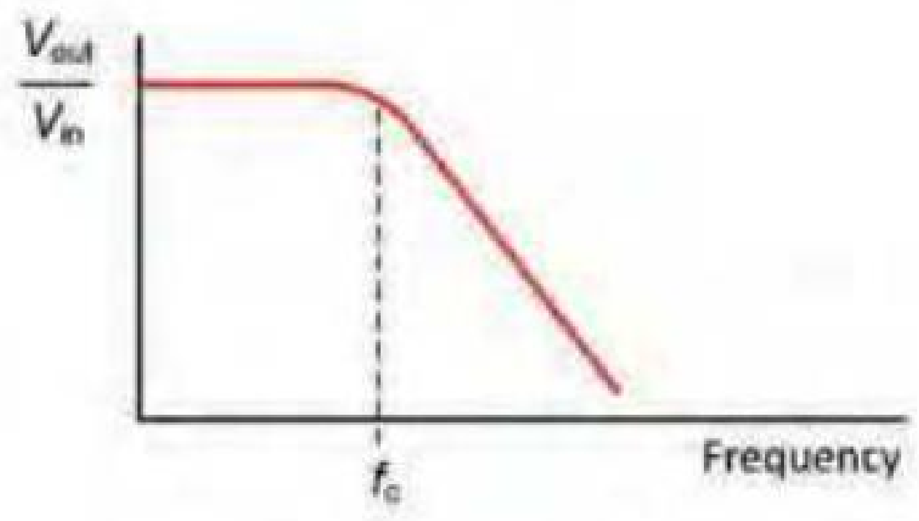


(c) Filtered analogue output

Fig.9.6. DAC input and output waveforms.


From
DAC output
Low-pass
filter
Filtered
analogue output

(a) Filtered DAC output



(b) Filter characteristic

Fig.9.7. Using a low-pass filter with a DAC.

Unfortunately, because of the range of resistance values required, the binary-weighted DAC becomes increasingly impractical for higher-resolution applications in intefrated circuits. Taking a 10-bit circuit as an example, and assuming that the basic value of R is 1kΩ, the binary weighted values would range from 1kΩ (bit-0) to 256kΩ (bit-9). To ensure high accuracy, these resistors will need to be close-tolerance types (±1%, or better).

A more practical arrangement uses a summing amplifier in which the input voltage to the operational amplifier is derived from an R-2R ladder, as shown in Fig.9.5(c). Note that only two resistance values are required and that they can be any convenient value provided that one value is double the other (it is relatively easy to manufacture matched resistances of close tolerance and high stability on an integrated circuit chip).

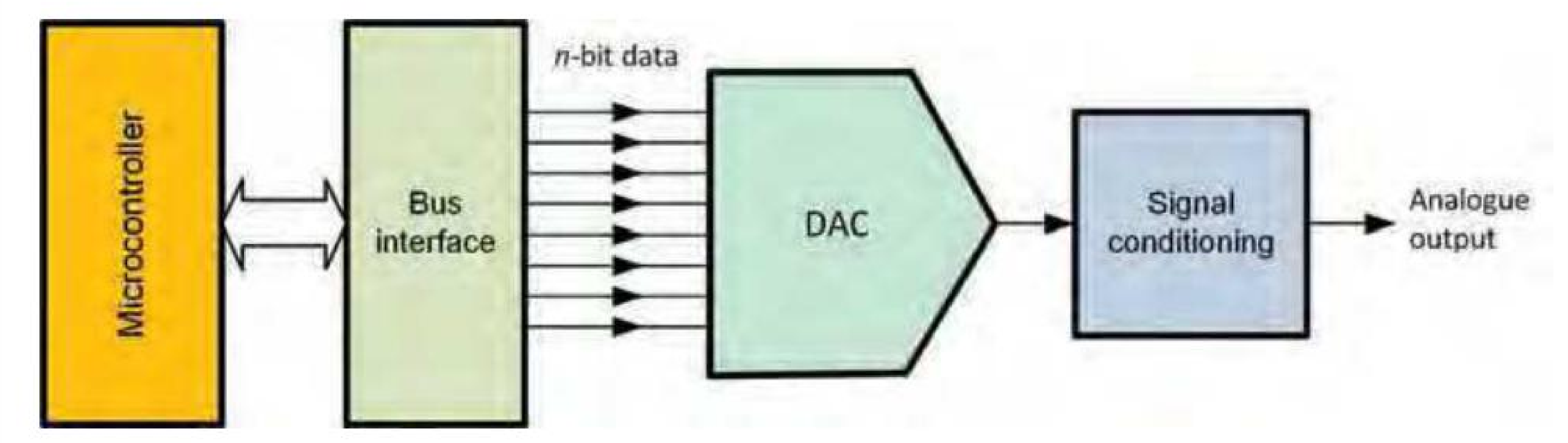

Microcontroller
Bus
interface
n-bit data
DAC
Signal
conditioning
Analogue
output

Fig.9.8. Connecting an external DAC to a microcontroller.

Fig.9.6 shows how a varying analogue output voltage can be produced by the DAC arrangements in Fig.9.5. The presence of discrete voltage steps in Fig.9.6(b) (rather than a smooth waveform) is undesirable for many applications, but this can be resolved by passing the output signal through an appropriately designed low-pass filter, as shown in Fig.9.7.

Fig.9.8 shows how an external DAC can be connected to a microcontroller. The bus interface can be based on the I^{2}C bus or on the faster SPI bus – most microcontrollers support both.

A practical add-on DAC

Fortunately, it is easy to add true DAC capability to any popular microcontroller board (including the Arduino or Raspberry Pi Pico boards that we met earlier). This is made even easier by using a low-cost DAC module based on an IC like the MCP4725 from Microchip (see Fig.9.9). This high-accuracy, single-channel, 12-bit buffered output DAC incorporates a non-volatile memory that will allow you to 'set and forget' the data sent to it. The chip's on-board precision output amplifier allows it to achieve a full-range (rail-to-rail) analogue output voltage swing. The chip incorporates an I^{2}C-compatible serial interface which can operate in standard (100kHz), fast (400kHz), or high-speed (3.4MHz) modes.

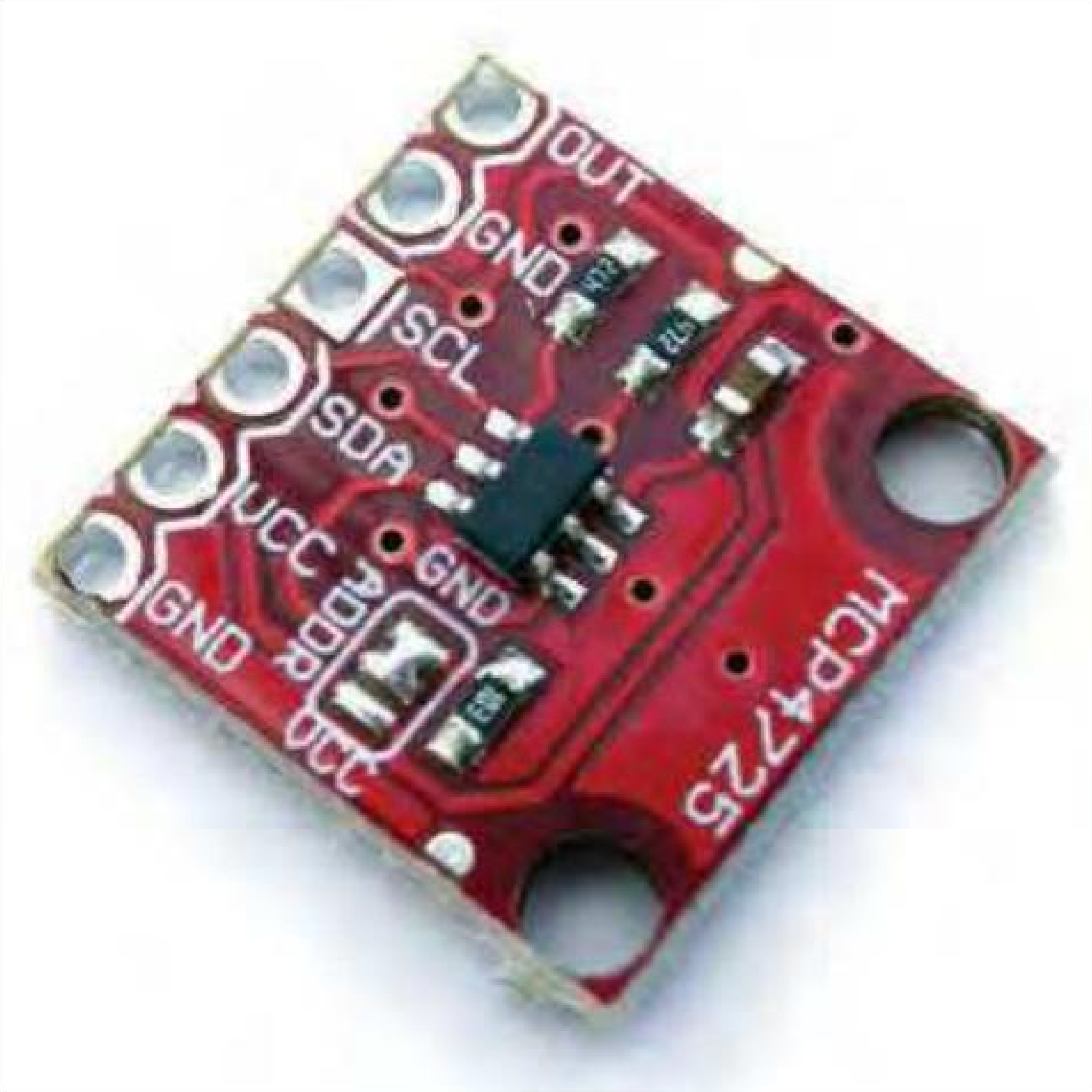

OUT
GND
SCL
SDA
VCC
GND
ADDR
GND
VCC
MCP4725

Fig.9.9. A low-cost DAC module based on the MCP4725.

The simplified internal architecture of the MCP4725 is shown in Fig.9.10. The chip's internal non-volatile memory retains the digital input value during power-off time, and the previously set DAC output will become available immediately after power-up. The network address selection pin (A0) can be taken high or low to facilitate address selection.

Fig.9.11 shows the circuit schematic of a simple buffered MCP4725 that can be used with an Arduino microcontroller. The output voltage of the 12-bit DAC can be set over the range 0 to +5V with

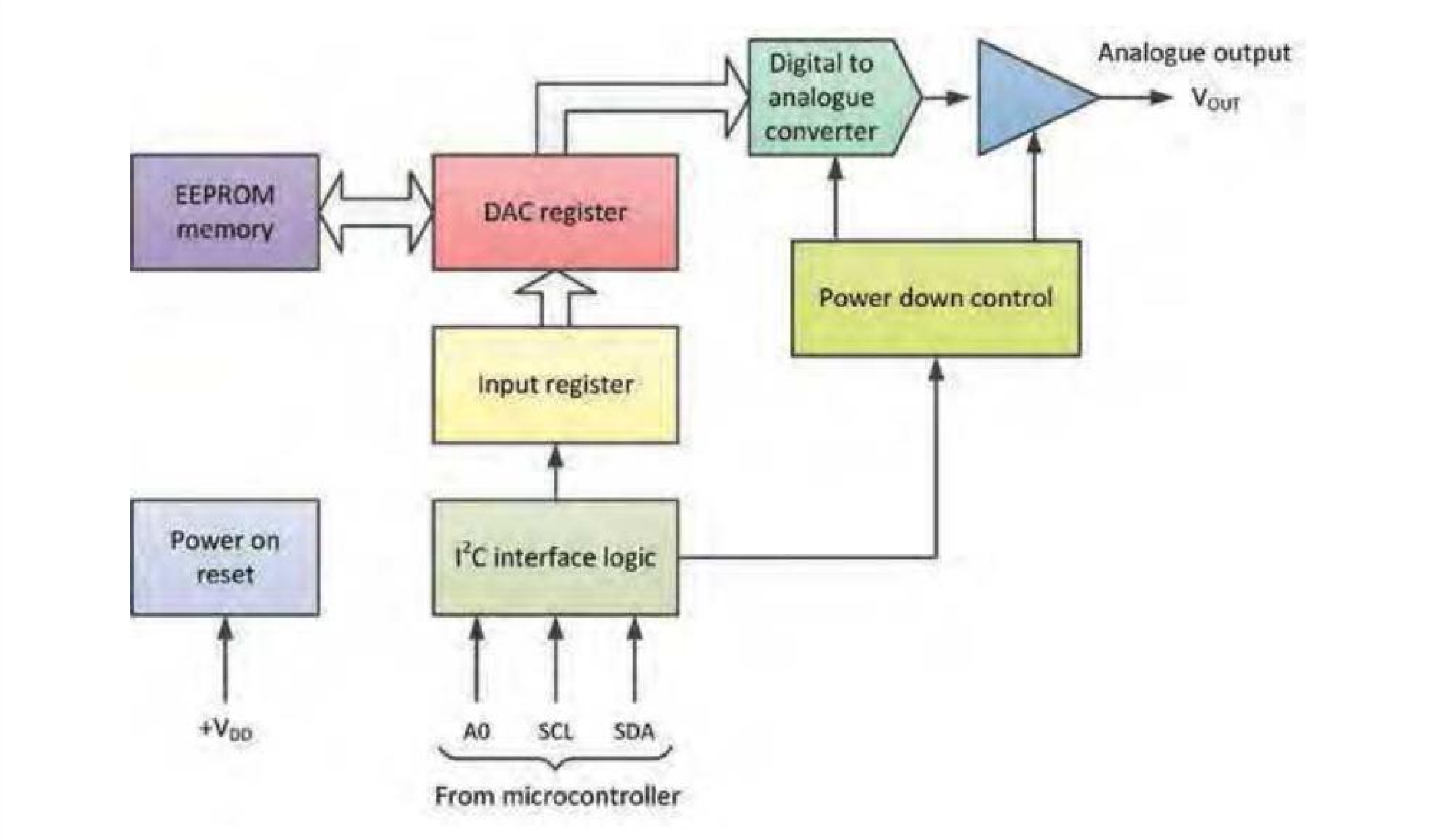

Digital to
analogue
converter
Analogue output
V_{OUT}
EEPROM
memory
DAC register
Power down control
Input register
Power on
reset
I^{2}C interface logic
$+V_{DD}$
A0
SCL
SDA
From microcontroller

Fig.9.10. Simplified internal architecture of an MCP4725 DAC.

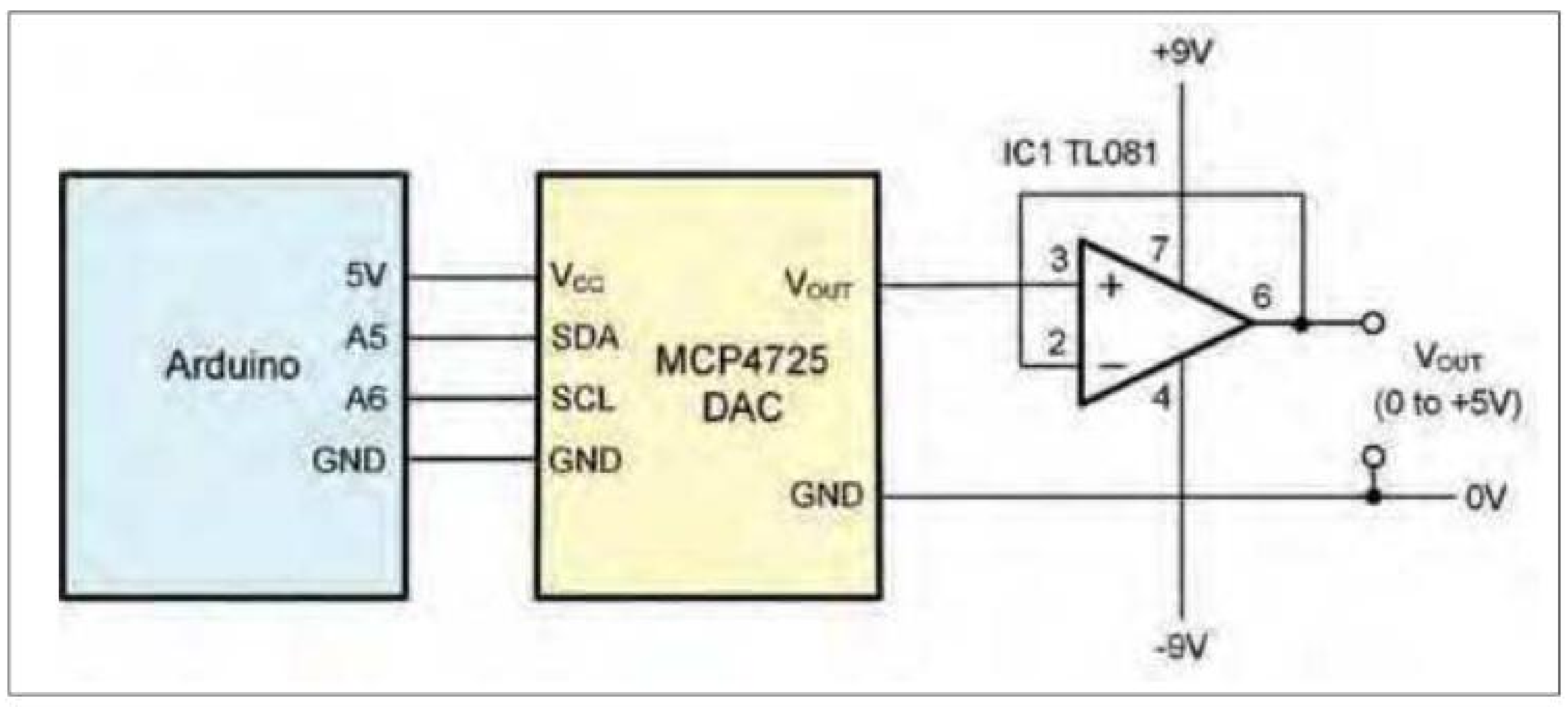


Fig.9.11. A simple buffered MCP4725 DAC with positive output.

4096 steps of 1.22mV. It is worth noting that the chip uses the positive supply voltage (V_{DD}) as its voltage reference. This is expedient but it means the supply rail should be accurate, stable and noise and hum free. For this reason, it is important to ensure that the 5V supply is well regulated and has suitably rated capacitors to reduce supply noise and ripple.

Coding the MCP4725

Pre-made library modules make DAC coding very straightforward, but before entering and testing your code you will need to locate and install the required library module (see Fig.9.12). To check the DAC's accuracy and linearity, the circuit in Fig.9.11 was tested by sending a series of data values (from 0 to 4095) to the chip and measuring the output voltage with an accurate bench voltmeter. The results are shown in Table 9.4.

Table 9.4 Measured output voltages for Fig.9.10

Binary code (base 2)	Hex (base 16)	Denary (base 10)	Output voltage (V)
000000000000	000	0	0
000000000001	001	1	0.00122
000000000010	002	2	0.00244
000000000011	003	3	0.00366
000000000100	004	4	0.00488
….	….	….	….
001100110010	332	818	0.99854
001100110011	333	819	1.00000
001100110100	334	820	1.00098
….	….	….	….
111111111101	FFD	4093	4.99634
111111111110	FFE	4094	4.99756
111111111111	FFF	4095	4.99878

Listing 9.1 is a complete sketch showing the code required to generate a simple staircase waveform based on the circuit in Fig.9.11. The code generates six voltage steps, from 0V to 5V, over a period of 24 seconds, as depicted in Fig.9.13. Note that the currently generated voltage is echoed back to the host computer using the serial monitor facility.

Listing 9.1 Code for the positive staircase function

```
// Positive staircase function generator
// 0 to 5V in 1V steps using the MCP4725
// Rob Tillaart's MCP4725 library is available from
// https://github.com/RobTillaart/MSP4725

#include "Wire.h"   // Libraries required
#include "MCP4725.h"

MCP4725 MCP(0x60);  // Can be 0x62 or 0x63

void setup()
{
  Serial.begin(115200);
  Serial.println("0 to 5V in 1V steps:");
  MCP.begin();
  MCP.powerOnWakeUp();
  MCP.setValue(0);
}

void loop()
{
  for (uint16_t i = 0; i < 6; i++)
  {
    // Rising steps
    Serial.print(i);
    Serial.println(" V");
    MCP.setValue(i * 819);
    delay(4000);  // 4s delay between steps
  }
}
```

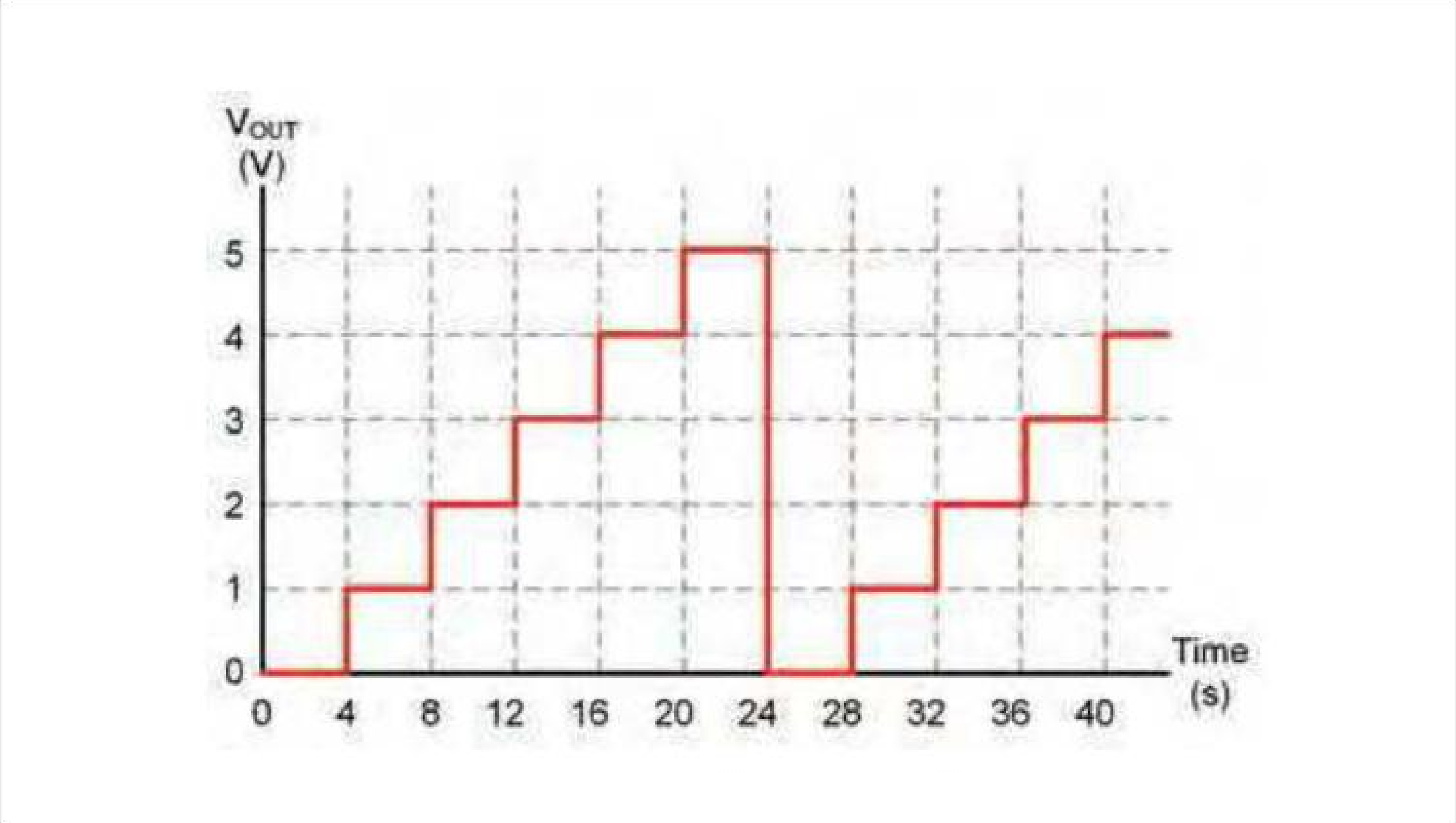
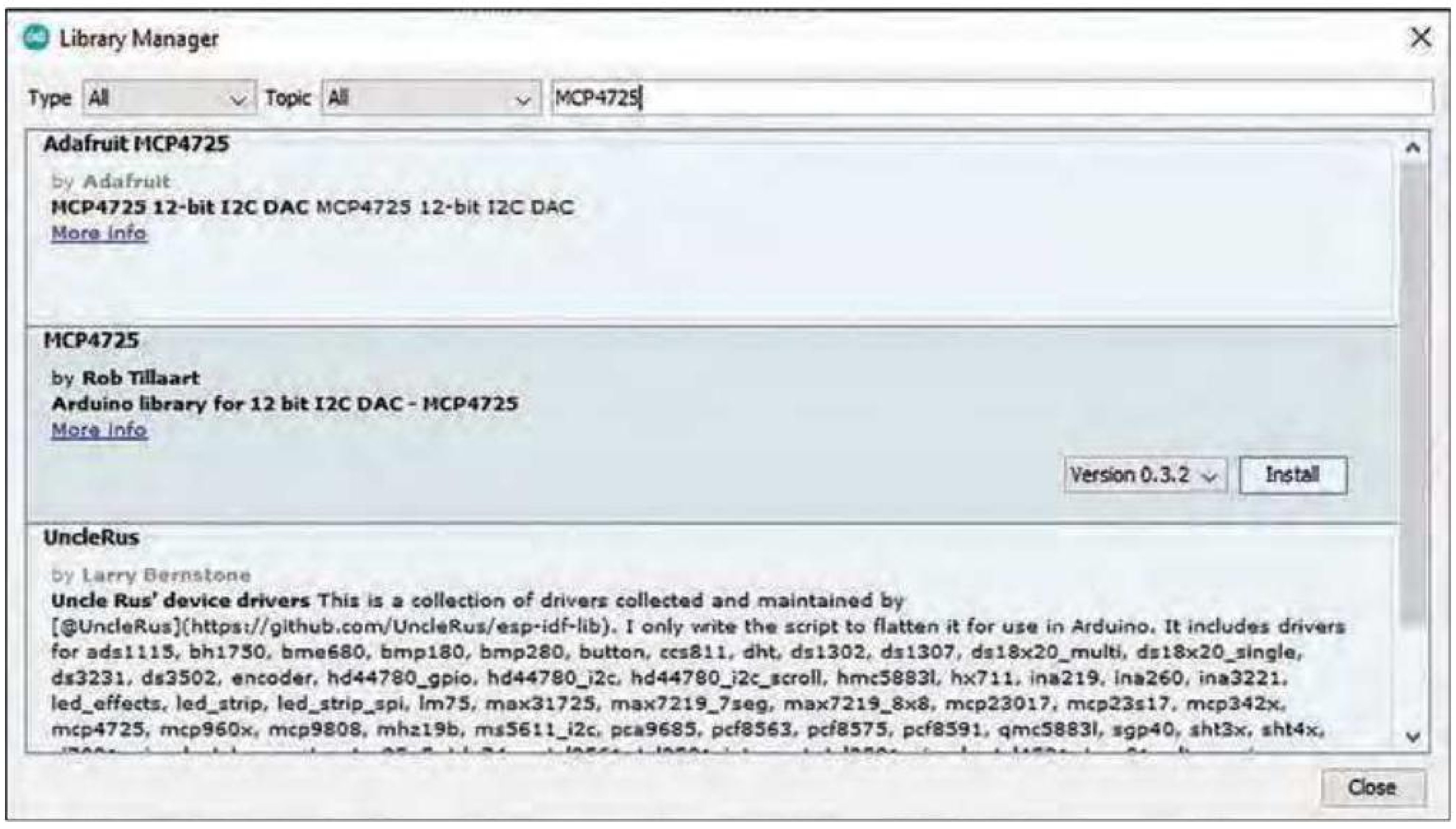


Fig.9.13. Positive staircase waveform produced by Fig.9.11 and Listing 9.1.



MCP4725

Adafruit MCP4725

by Adafruit

MCP4725 12-bit I2C DAC MCP4725 12-bit I2C DAC

MCP4725

by **Rob Tillaart**

Arduino library for 12 bit I2C DAC - MCP4725

UncleRus

by Larry Bernstone

Uncle Rus' device drivers This is a collection of drivers collected and maintained by [@UncleRus](https://github.com/UncleRus/esp-idf-lib). I only write the script to flatten it for use in Arduino. It includes drivers for ads1115, bh1750, bme680, bmp180, bmp280, button, ccs811, dht, ds1302, ds1307, ds18x20_multi, ds18x20_single, ds3231, ds3502, encoder, hd44780_gpio, hd44780_i2c, hd44780_i2c_scroll, hmc5883l, hx711, ina219, ina260, ina3221, led_effects, led_strip, led_strip_spi, lm75, max31725, max7219_7seg, max7219_8x8, mcp23017, mcp23s17, mcp342x, mcp4725, mcp960x, mcp9808, mhz19b, ms5611_i2c, pca9685, pcf8563, pcf8575, pcf8591, qmc5883l, sgp40, sht3x, sht4x,

Fig.9.12. Using the Arduino's Library Manager to locate and install the MCP4725 library.

Producing a negative output voltage

If a negative output voltage is required, the circuit can be easily modified, as shown in Fig.9.14. The required code is shown in Listing 9.2 and the code generates six voltage steps, from 0V to –5V, over a period of 24 seconds, as shown in Fig.9.15. As with the previous listing, the currently generated voltage is echoed back to the host computer using the serial monitor facility.

Listing 9.2 Code for the negative staircase function

```
// Negative staircase function generator
// 0 to -5V in -1V steps using the MCP4725
// Rob Tillaart's MCP4725 library is available
// from https://github.com/RobTillaart/MSP4725

#include "Wire.h"   // Libraries required
#include "MCP4725.h"

MCP4725 MCP(0x60);  // Can be 0x62 or 0x63

void setup()
{
  Serial.begin(115200);
  Serial.println("0 to -5V in 1V steps:");
  MCP.begin();
  MCP.powerOnWakeUp();
  MCP.setValue(0);
}

void loop()
{
  for (uint16_t i = 0; i < 6; i++)
  {
    // Falling steps
    Serial.print(i);
    Serial.println(" V");
    MCP.setValue(4095 - (i * 819));
    delay(4000);  // 4s delay between steps
  }
}
```

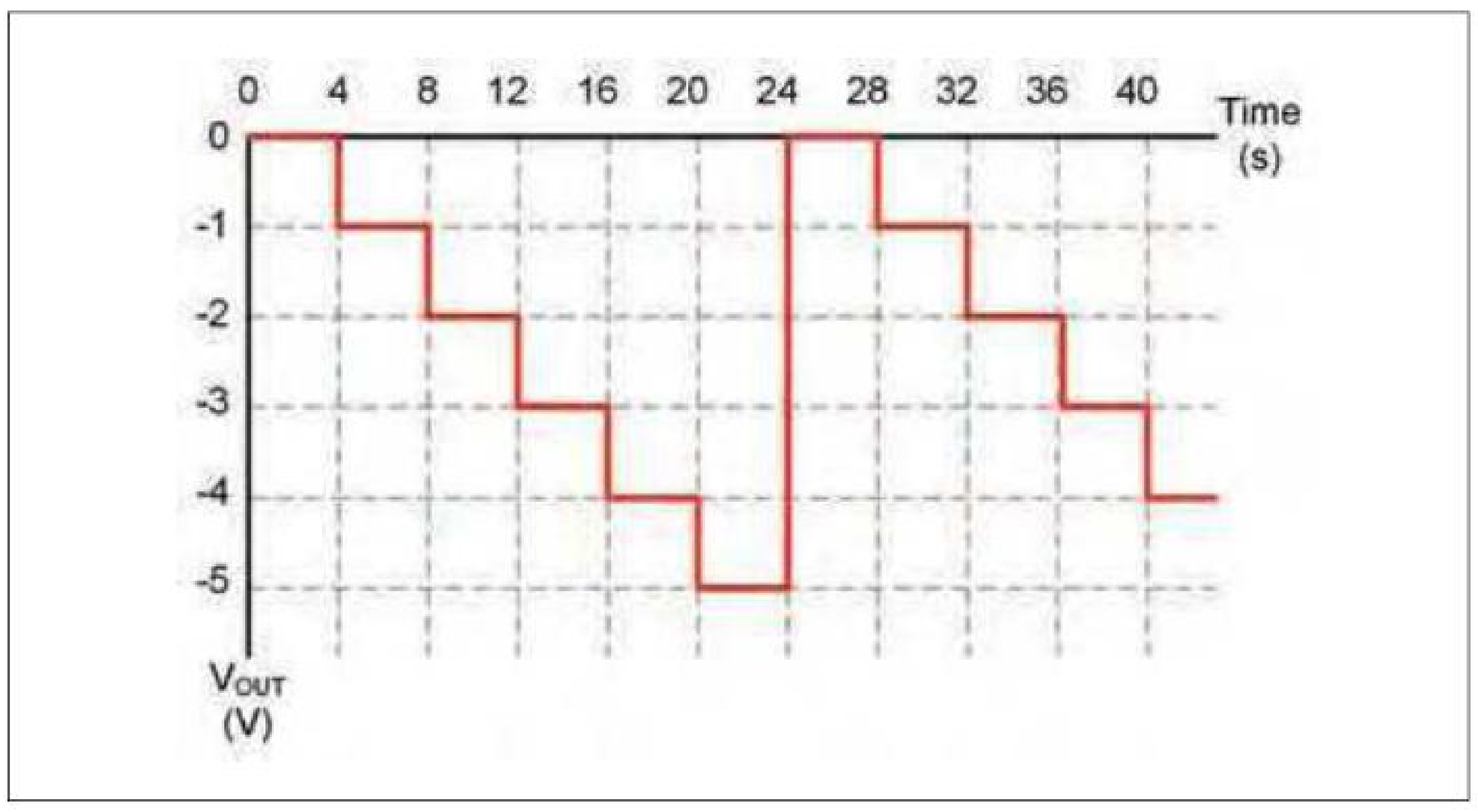


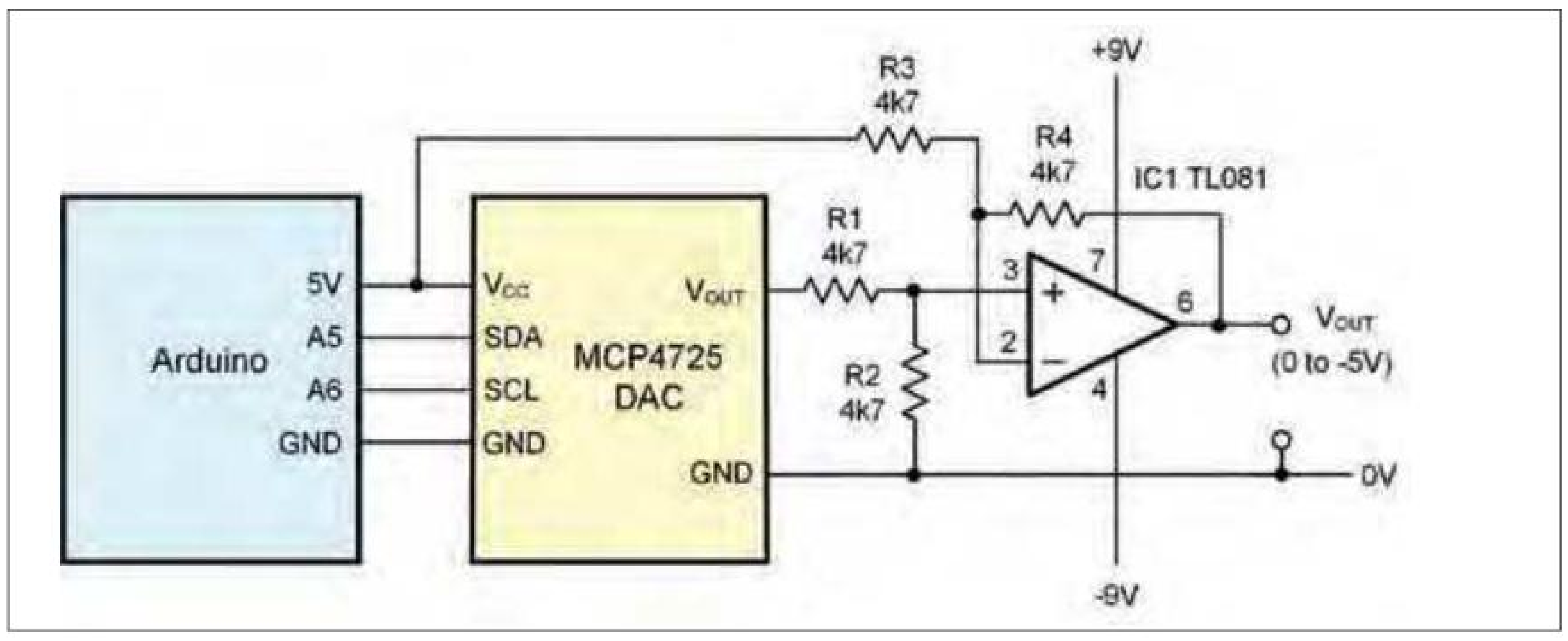
Fig.9.15. Negative staircase waveform produced by Fig.9.14 and Listing 9.2.

Dual output arrangement

If both positive and negative voltages are required, the dual output circuit shown in Fig.9.16 can be employed. Here, IC1 operates as a unity-gain inverter (note that the positive output remains unbuffered in this arrangement).

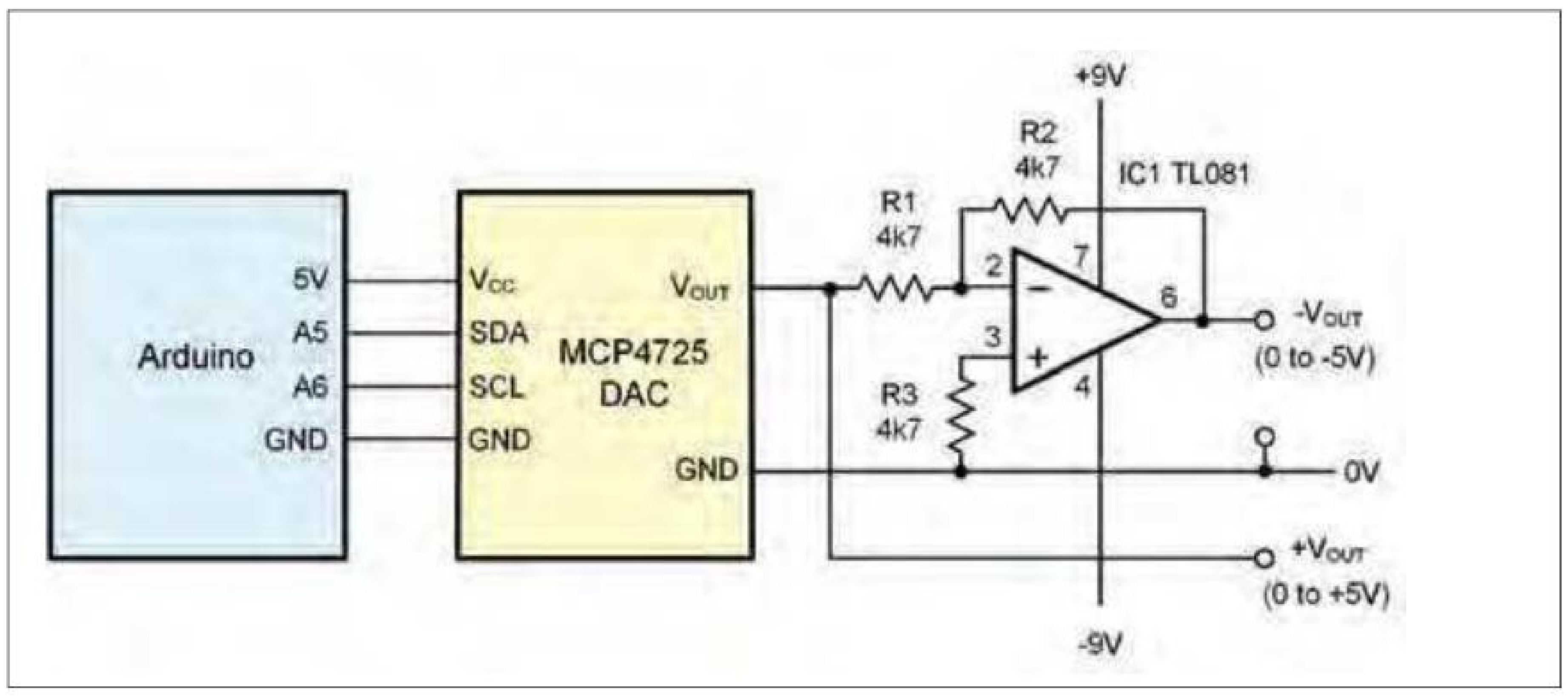
Boosting the output current

In the simple buffered arrangements shown in Fig.9.11 and 9.14, the output current supplied to a load should be limited to 20mA. Where necessary, the current drive capability can be increased by adding an emitter-follower stage (TR1), as shown in Fig.9.17 and Fig.9.18 for positive and negative outputs respectively. Both circuits can deliver load currents of up to about 0.25A, but a small heatsink may be required for load currents of more than 100mA.



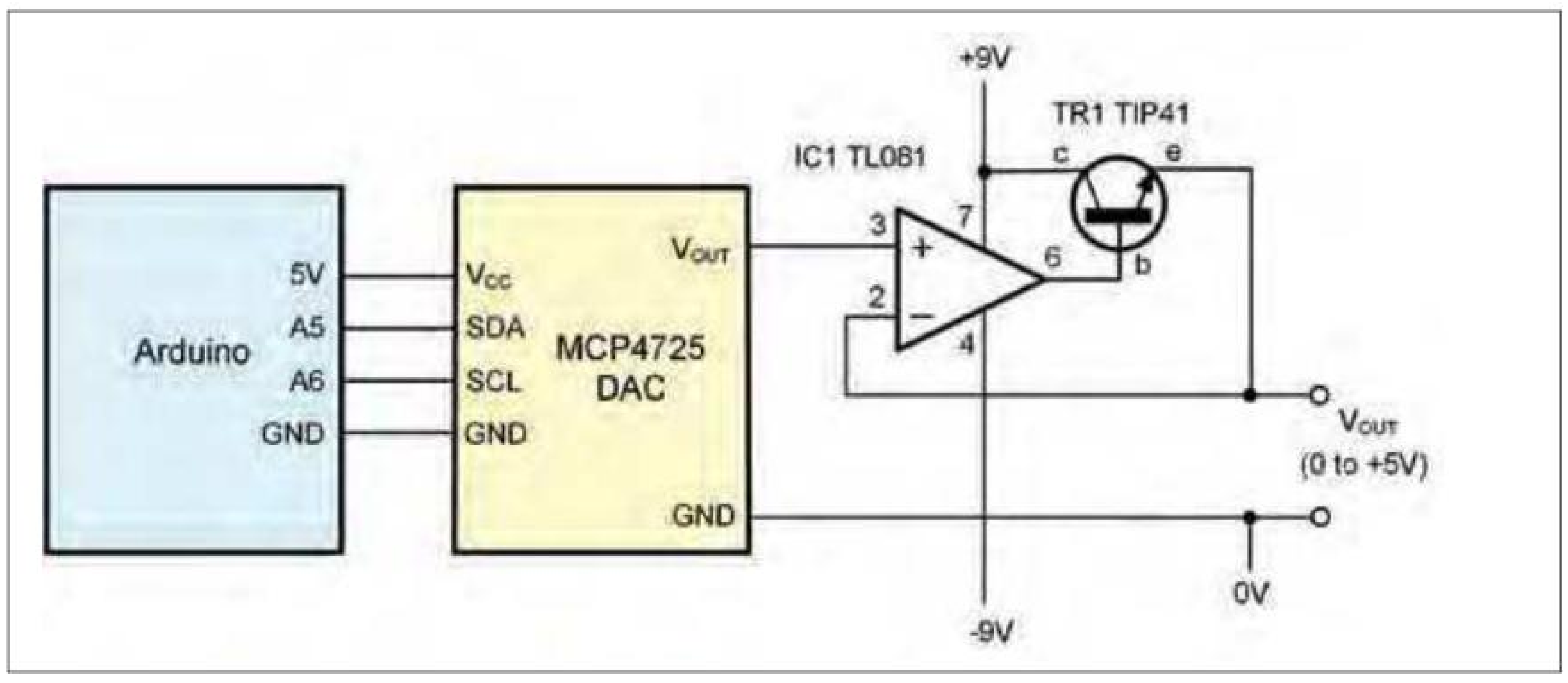
Arduino
5V
A5
A6
GND
Vcc
SDA
SCL
GND
VOUT
GND
MCP4725 DAC
R3 4k7
R4 4k7
R1 4k7
R2 4k7
IC1 TL081
+9V
-9V
3
2
7
6
4
VOUT (0 to -5V)
0V

Fig.9.14. Buffered MCP4725 DAC with negative output.



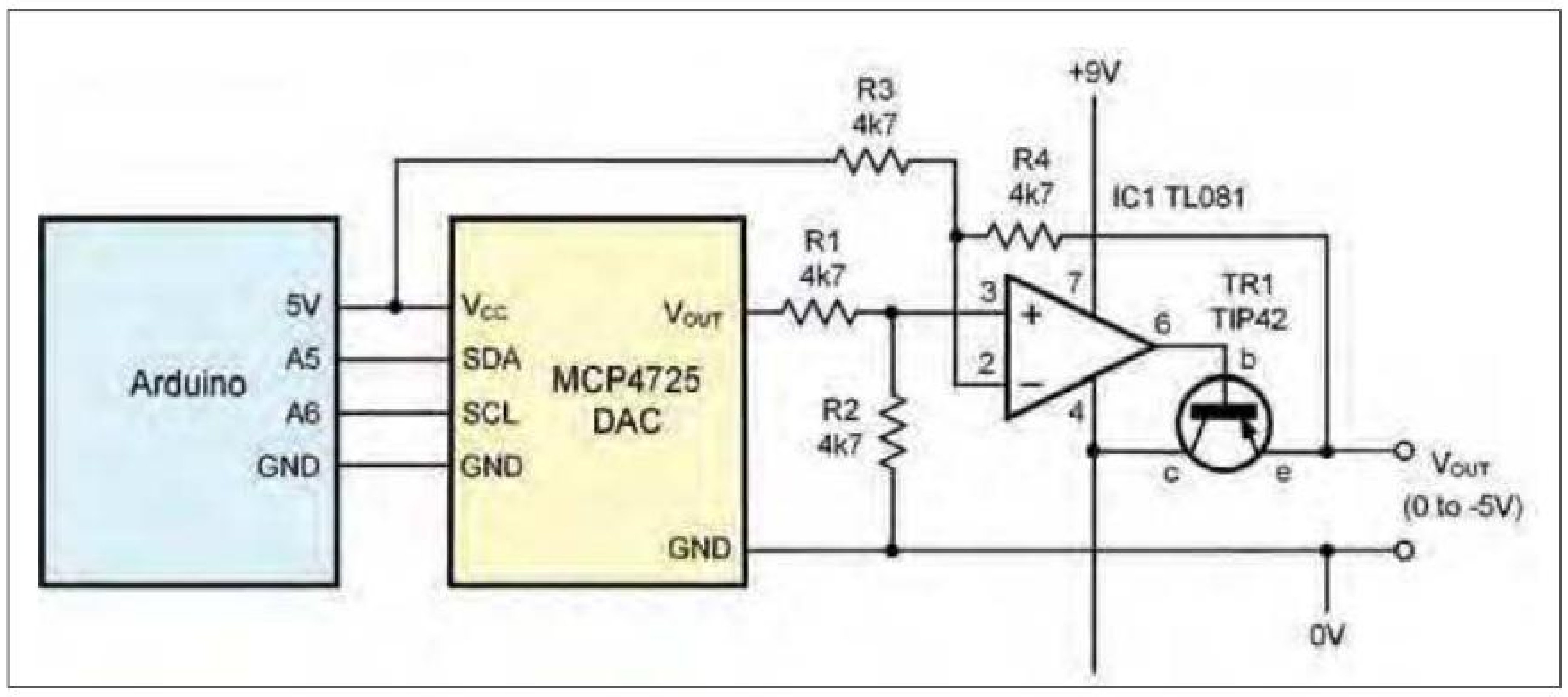
Arduino
5V
A5
A6
GND
Vcc
SDA
SCL
GND
VOUT
GND
MCP4725 DAC
R1 4k7
R2 4k7
R3 4k7
IC1 TL081
+9V
-9V
2
3
7
6
4
-VOUT (0 to -5V)
0V
+VOUT (0 to +5V)

Fig.9.16. Dual output DAC (positive and negative outputs are simultaneously available).



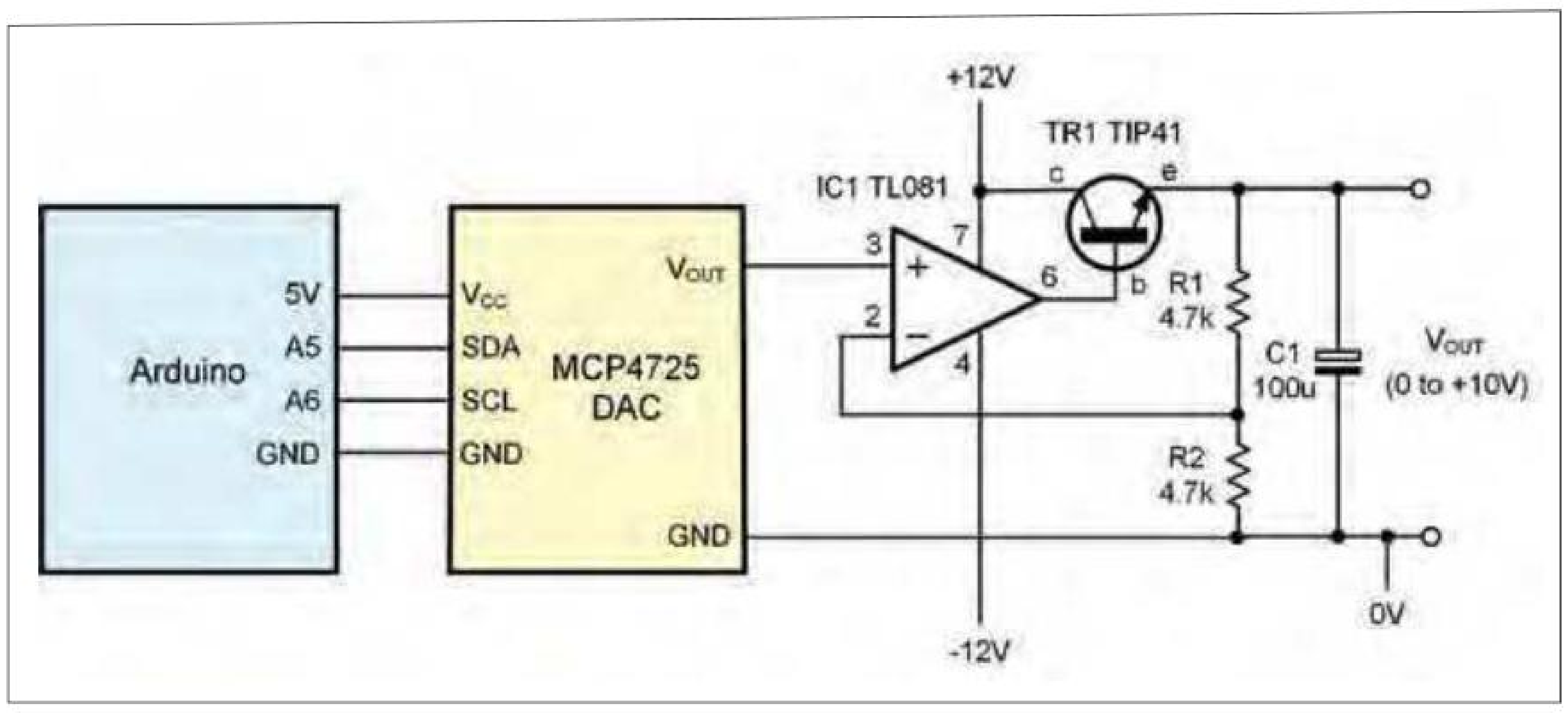
Arduino
5V
A5
A6
GND
Vcc
SDA
SCL
GND
VOUT
GND
MCP4725 DAC
IC1 TL081
TR1 TIP41
c
e
b
+9V
-9V
3
2
7
6
4
VOUT (0 to +5V)
0V

Fig.9.17. Current-boosted DAC with positive output.



Arduino
5V
A5
A6
GND
Vcc
SDA
SCL
GND
VOUT
GND
MCP4725 DAC
R3 4k7
R4 4k7
R1 4k7
R2 4k7
IC1 TL081
TR1 TIP42
b
c
e
+9V
3
2
7
6
4
VOUT (0 to -5V)
0V

Fig.9.18. Current-boosted DAC with negative output.



Arduino
5V
A5
A6
GND
Vcc
SDA
SCL
GND
VOUT
GND
MCP4725 DAC
IC1 TL081
TR1 TIP41
c
e
b
R1 4.7k
R2 4.7k
C1 100u
+12V
-12V
3
2
7
6
4
VOUT (0 to +10V)
0V

Fig.9.19. Basis of the author's digitally controlled low-voltage 0.25A DC power supply.

Table 9.5. Going Further with Exploring microcontroller digital-to-analogue conversion

Topic	Source	Notes
PWM	Introduction to PWM: **https://bit.ly/pe-aug22-pwm** This Texas Instruments application report – **https://bit.ly/pe-aug22-ti** – describes a method for using PWM as a DAC for its series of digital signal controllers; includes a detailed explanation of analogue filter design.	The website provides links to a number of practical projects that use PWM techniques.
MCP4725 DAC	The MCP4725's datasheet can be downloaded from Microchip's website by going to: **https://bit.ly/pe-aug22-mcp** Useful guide to the MCP4725: **https://bit.ly/pe-aug22-best**	
Arduino	*Electronics Teach-In 8* from *PE* / Electron Publishing: **http://bit.ly/pe-apr21-ks2-7** Comprehensive guide to the Arduino – this popular series introduces hardware and software and also features a range of practical projects with different levels of complexity.	
Arduino IDE	The Arduino's integrated development environment (IDE) can be downloaded from: **https://bit.ly/pe-dec21-ard2**	Versions available for Windows, Linux and macOS
Raspberry Pi Pico	The official Raspberry Pi Pico guide is *Get Started with Micropython on Raspberry Pi Pico* by Gareth Halfacre and Ben Everard (ISBN 978-1-912-04786-4). Provides an introduction to the Pico and using Thonny, describes simple beginner projects. *Programming the Pico – Learn Coding and Electronics with the Raspberry Pi Pico* (ISBN 979-8-464-88217-1) an excellent introductory book from respected author Simon Monk.	These two books are intended for complete beginners and they both assume little previous knowledge of electronics and coding.

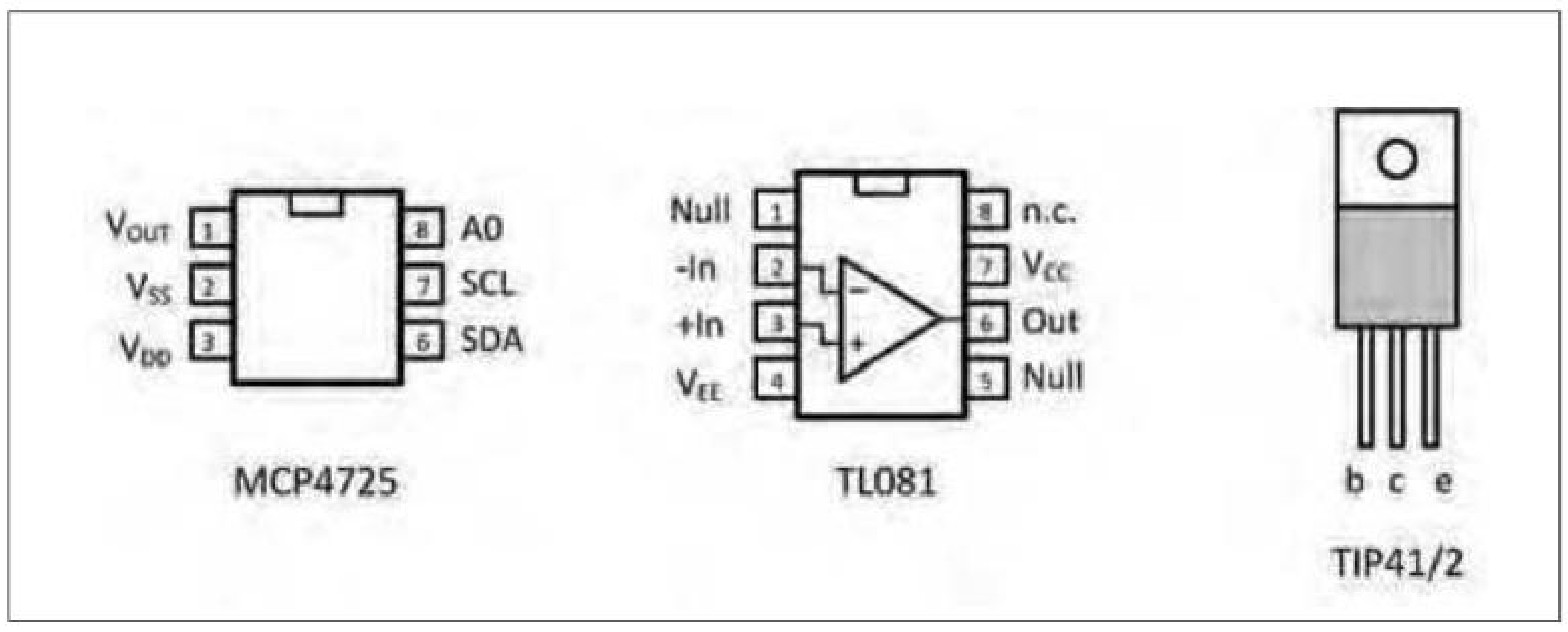


Fig.9.20. Semiconductor pin connections.

Increasing the output voltage

If more than 5V is required from the DAC, the output voltage can be increased by incorporating gain in the buffer stage, as shown in Fig.9.19. This circuit forms the basis of the author's digitally controlled low-voltage power supply. Note the use of +12V and –12V supply rails (the positive rail should be rated at about 1A while the negative rail is less demanding at 100mA, or so). Pin connections for the semiconductors are shown in Fig.9.20.

Going further

Table 9.5 in this section details a variety of sources (books and online links) that will help you locate the component parts and further information that will enable you to understand the use of PWM, as well conventional DAC devices. It also provides links to underpinning knowledge and manufacturers' data sheets.

CRICKLEWOOD ELECTRONICS
Established 1981

Your best bet since MAPLIN
Chock-a-Block with Stock

Visit: www.cricklewoodelectronics.com
Or phone our friendly knowledgeable staff on 020 8452 0161
Components • Audio • Video • Connectors • Cables
Arduino • Test Equipment etc, etc

Visit our Shop, Call or Buy online at:
www.cricklewoodelectronics.com
020 8452 0161
Visit our shop at:
40-42 Cricklewood Broadway
London NW2 3ET

Max's Cool Beans

By Max the Magnificent

Flashing LEDs and drooling engineers – Part 30

I am a huge fan of Douglas Adams (RIP). In addition to *The Hitch Hiker's Guide to the Galaxy*, I really like his tall tales about Dirk Gently, who bills himself as a 'holistic detective' who makes use of 'the fundamental interconnectedness of all things.' The reason I mention this here is that I'm currently experiencing a lot of fundamental interconnectedness myself, which is unfortunate because I don't have a speech prepared and I have nothing suitable to wear.

A future replete with robots

As you will doubtless (well, hopefully) be aware from previous *Cool Beans* columns, over the past few months, your humble narrator and his friend Steve Manley have been working on an animatronic robot head. In reality, Steve has done the bulk of the grunt work while I've focused my attention on offering silly suggestions (I like to play to my strengths). As a result of all this activity, I've been spending a lot of time thinking about things like robots and motors and sensors.

My wife (Gina the Gorgeous) often asks me how long it will be before we will be able to buy a robot to help her with her household chores. I fear she's been watching too many science fiction films with me because she's thinking of a humanoid-shaped incarnation that will be able to perform activities like crouching down and picking things up and putting them away, loading and emptying the dishwasher, similarly with the washing machine and dryer (including ironing and folding the clothes), watering the flowers in the pots and baskets on our front and back porches, accompanying her to the grocery store where she can say things like 'get me two large cans of diced tomatoes and meet me at the meat counter,' and so on and so forth.

Not wishing to raise her hopes with unrealistic expectations, I've been telling Gina that it is going to be quite some time before this sort of thing comes to pass. In my heart of hearts, I've feared that we may not be around to see it. At the same time, having recently watched the post-apocalyptic science fiction thriller film *Mother Android* (**https://bit.ly/3N3r8tz**), I've also been afraid that we might live long enough to regret wanting to live long enough to see it (you really don't want to peer into what I laughingly call my mind).

For example, although they are incredibly clever, the humanoid robots from Boston Dynamics that can do things like dance (**https://bit.ly/3y2cezp**) and perform gymnastics (**https://bit.ly/3OpPkXY**) are more along the lines of uber-expensive proof-of-concept creations affordable only to mega-corporations and nation states. Having said this, I've seen some things over the past couple of weeks that lead me to believe a future replete with robots may be closer than we think.

Robots, motors and sensors, Oh my!

When you come to think about it, we see and hear (no pun intended) a lot of news about the latest and greatest in visual and audio sensors giving machines the ability to see and hear, including things like object detection and recognition and the ability to recognise sounds like glass breaking and to understand and respond to natural human speech. There are also developments in gustatory and olfactory sensors that will afford machines the ability to detect and identify tastes and smells. Less discussed, however, are tactile sensors that will give machines a sense of touch, but this too may be poised to change.

For example, I just read an interesting article on *Science Alert* about how scientists have developed a 'living skin' for robots (**https://bit.ly/3tJyXxs**). Looking really icky (if you'll forgive my talking technical), this 'material' is water repellent, self-healing, and has a texture just like human skin, which is perhaps not too surprising because it's actually made out of human skin cells.

By some strange quirk of interconnectedness fate, I was recently chatting with Keith McMillen, who is the founder and chief technical officer (CTO) of a company called Bebop Sensors (**https://bit.ly/3N3m4p3**). Of particular interest to us here, Keith has developed a smart sensing fabric that is used to create a skin-like covering called Bebop RoboSkin, which can provide humanoid robots with tactile awareness (Fig.1).

The amazing thing is that this tactile awareness is claimed to exceed the capabilities of human beings with respect to spatial resolution and sensitivity. In the example shown, there are 80 taxel

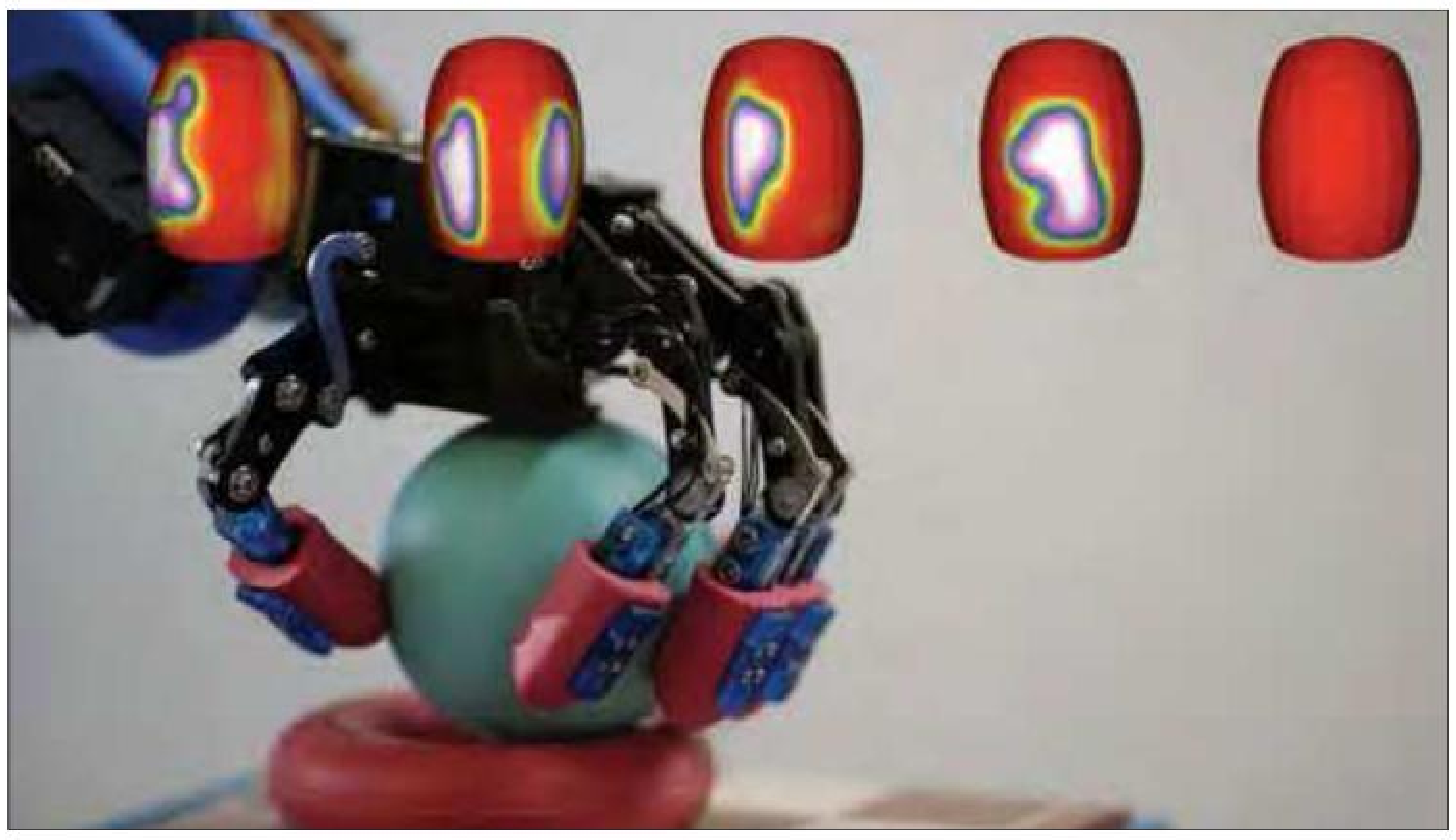


Fig.1. Robot hand equipped with BeBop RoboSkin picking up a ball (Image: Bebop Sensors).

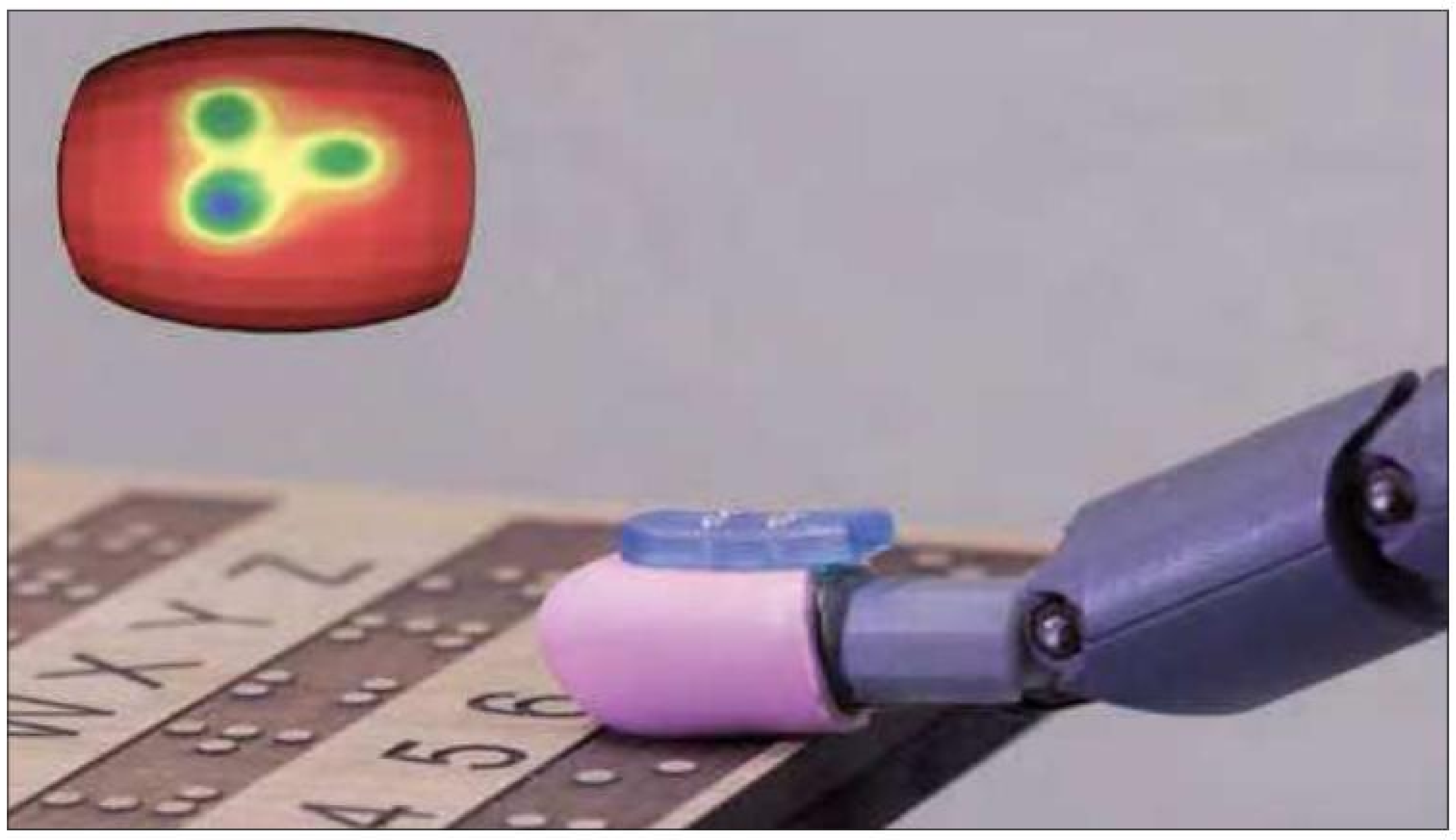


Fig.2. Robot finger equipped with BeBop RoboSkin reading Braille (Image: Bebop Sensors).

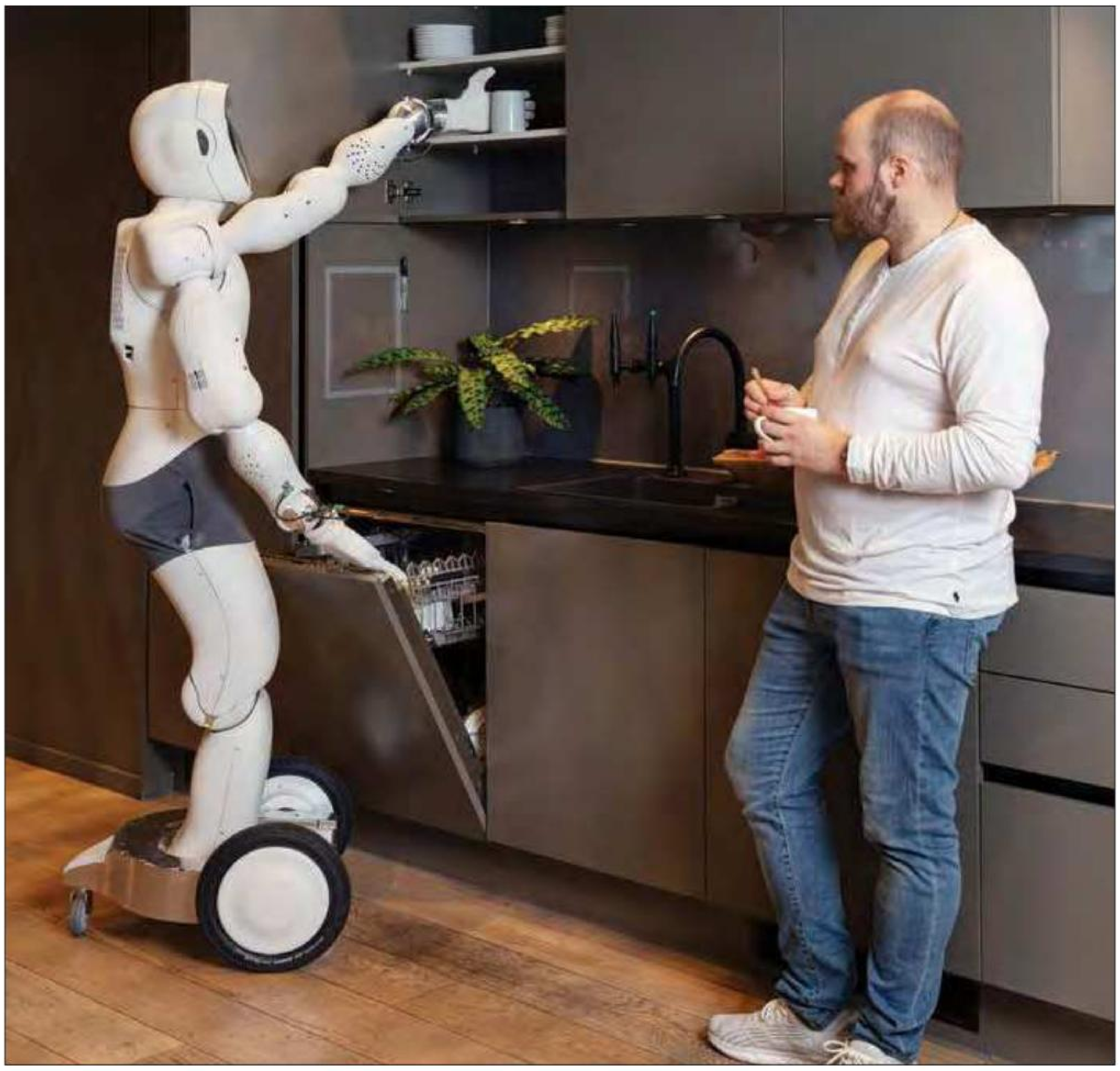


Fig.3. Meet EVE, the humanoid research robot (Image: Halodi Robotics)

sensors (think 'tactile pixels') in each fingertip, presented in an array with a 2mm x 3mm pitch. According to Keith, humans have about a 4mm pitch with respect to the nerves in our fingertips.

I must admit that I was somewhat doubtful when I first heard this. My human-centric knee-jerk reaction is to think that I am the end-result of billions of years of evolution that's left my biological sensors 'state-of-the-art,' as it were. However, Keith says that there's a simple way to verify his claims. If you take two plastic knitting needles with slightly curved tips, hold them side by side, close your eyes, and touch the ends of the knitting needles to the tip of one of your fingers, it's only when the distance between the points of the needles is 4mm or more that you can distinguish them as being separate.

Have you ever tried to see (again, no pun intended) if you could read Braille? The first version of this tactile writing system, which is used by people who are visually impaired, was developed in 1824 by a 15-year-old called Louis Braille in France who lost his sight as a result of a childhood accident.

Braille characters are formed using a combination of six raised dots arranged in a 3 × 2 matrix, which is called the 'braille cell.' This matrix offers 64 different combinations, which can be used to represent alphanumeric characters and punctuation marks (a cell with no dots is equivalent to a space).

I once had the opportunity to run one of my favorite fingers over a book in Braille, and I simply could not distinguish the number and locations of the raised dots under my dexterous digit. I understand that I could learn to do so with practice, but I remember being surprised by how little I could sense. This all came back to me when Keith showed me a picture of a robot finger equipped with BeBop RoboSkin reading Braille (Fig.2).

Robots, robots everywhere...

I have friends at a company called Immervision (**https://bit.ly/3QxxOTy**). They are working on the cutting edge of machine vision, developing cameras that combine ultra-wide-angle lenses with high-resolution sensors and sophisticated de-warping, image-stitching, and machine vision software based on artificial intelligence (AI), machine learning (ML) and deep learning (DL).

My friends were telling me that one of their current projects is with another company called Halodi Robotics (**https://bit.ly/39vFEg5**). You can probably guess what Holodi is working on. Literally yesterday as I pen these words, I was chatting with Dr. Nicholas Nadeau, who is Halodi's CTO.

The folks at Holodi have created an awesome autonomous android called Eve, which (or should I say 'who'?) is capable of performing some of the most dexterous of human tasks. Operating with human strength and speed, Eve is capable of crouching down, reaching up, and using 'her' hands to open doors, push buttons, and manipulate objects (Fig.3).

One of the reasons the guys and gals at Holodi are working with the chaps and chapesses at Immervision is that, in the event that Eve runs into a situation she can't handle, she can use the internet to call out to her human companions, who can use virtual reality goggles and haptic solutions to establish a telepresence connection – seeing through Eve's eyes and controlling her hands and body, and Immervision's ultra-wide-angle camera subsystems are ideal for this sort of application.

It's easy to think of all sorts of applications for Eve-type robots. With their ability to open doors and control elevators, tasks like patrolling buildings at night and stocking supermarket shelves spring to mind. Also, helping nurses by bringing drinks and meals to hospital patients. And, in the fullness of time, helping around the home (I'm not going to show Gina this article because she will never stop asking, 'when?').

One of the interesting things Nicholas told me is that they are doing a lot of work designing their own specialised motors that provide the optimal combination of speed and torque for their particular application. In turn, this caused me to return my attention to the motors powering my own animatronic appendage.

Motoring along

Have you ever thought to yourself, 'What is a motor?' I have. I wish I hadn't. It turns out that this is a topic of mindboggling complexity. As a start, although the terms are often used interchangeably, engines and motors are not necessarily the same thing. By one definition, engines convert chemical fuel into mechanical force by means of combustion, while motors (a.k.a. 'electrical motors') transform electrical energy into mechanical energy.

Having said this, some people define a motor as being 'a device that consumes energy in one form and converts it into motion or mechanical work.' By this definition, engines would form a sub-category of motors. There are also molecular motors – both natural (biological) and artificial (molecular machines) – that are essential agents in living organisms. If you are interested in learning more about these little rascals, I would strongly recommend the book *Life's Ratchet: How Molecular Machines Extract Order from Chaos* by Peter Hoffmann (**https://amzn.to/3tK6B6r**).

For the purposes of our animatronic noggin, we are interested in electrical

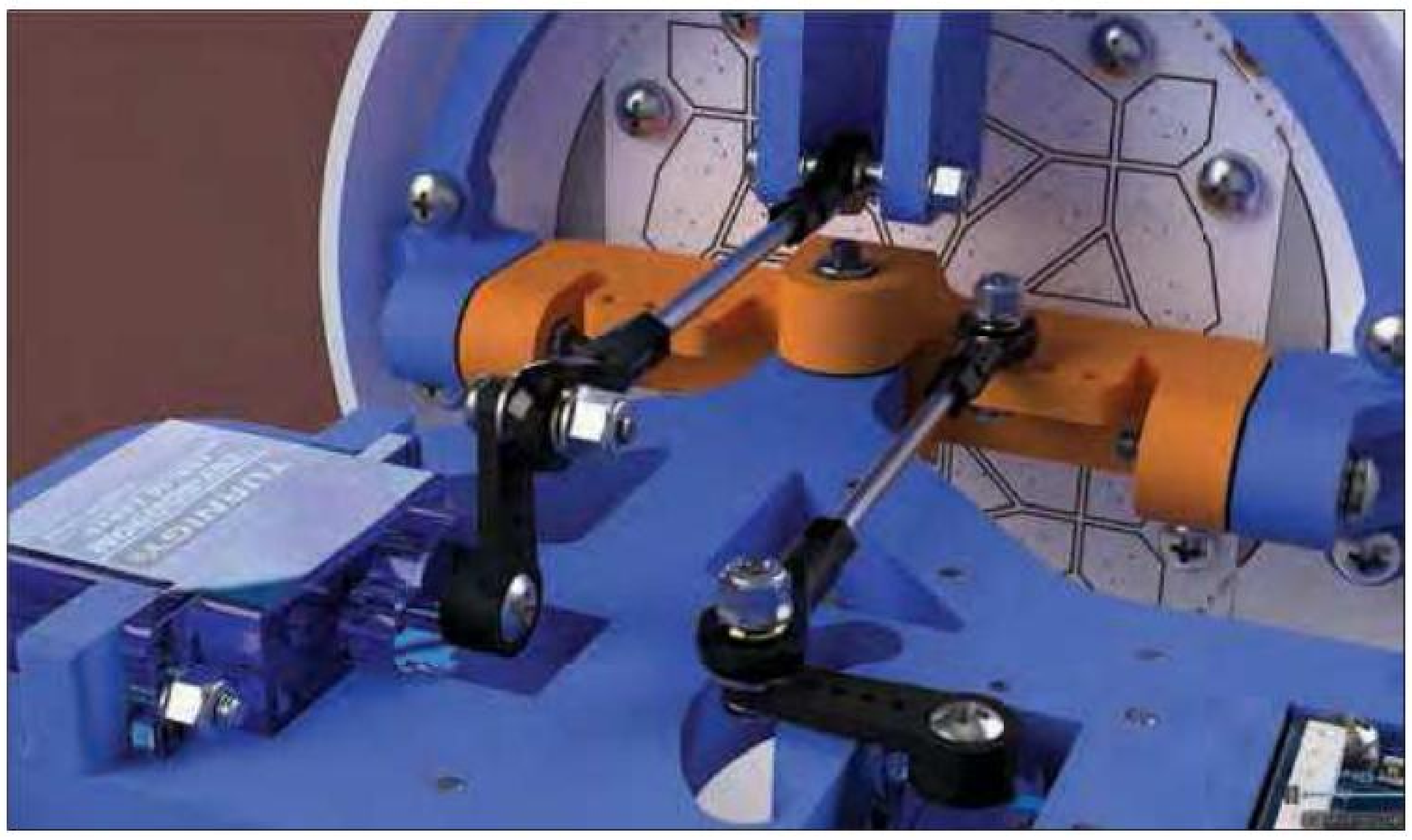


Fig.4. (above) Each animatronic eye is controlled by two 9g servos (Image: Steve Manley)

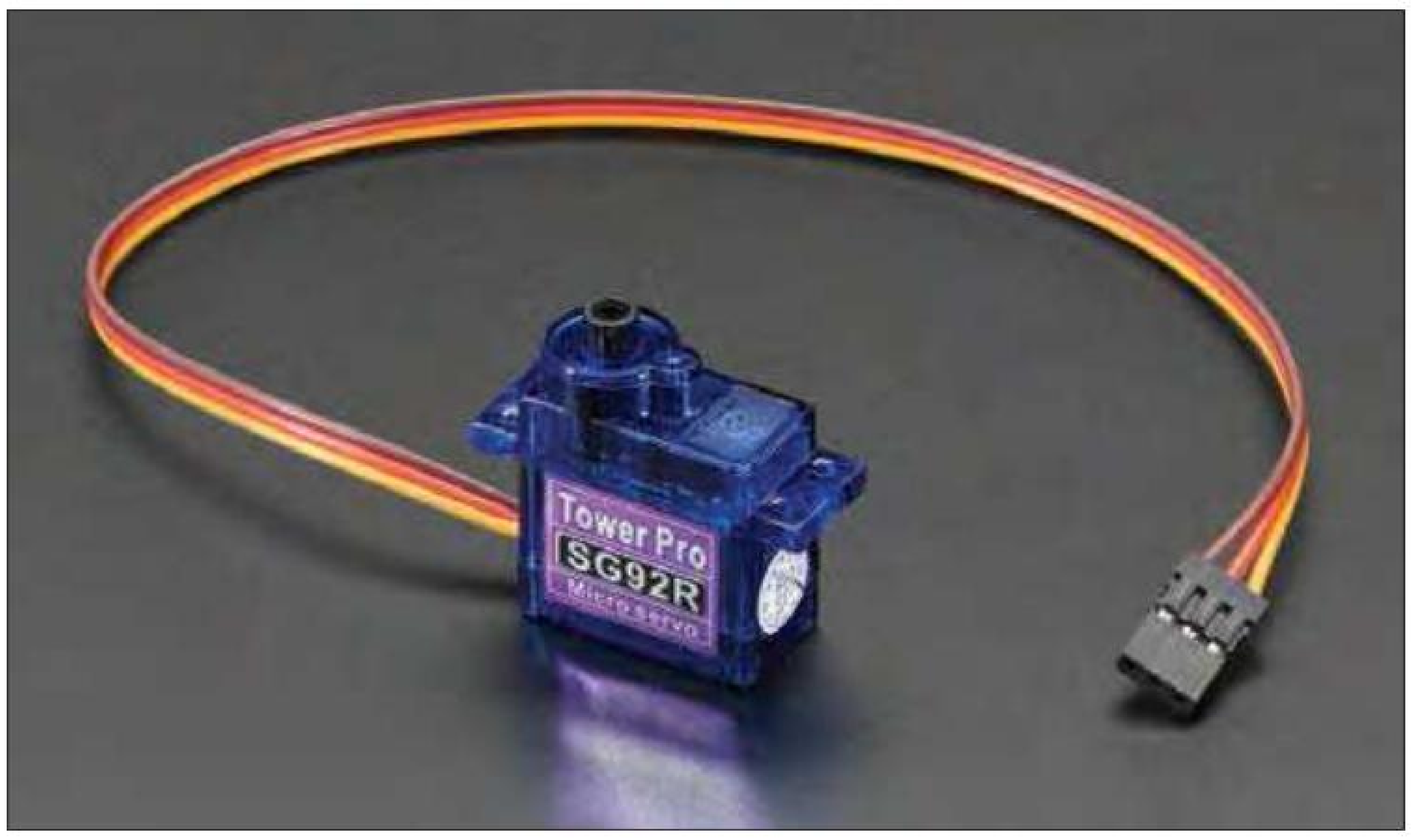


Fig.5. Tower Pro SG92R Micro Servo (Image: Adafruit)

motors, but there are so many different ways to 'slice and dice' this topic that it makes your (human and animatronic) heads spin. For example, we might divide things into rotary motors and linear motors, where the latter is essentially any electric motor that has been 'unrolled' so that, instead of producing torque (rotation), it produces a straight-line force along its length.

I must admit that, when I originally started to contemplate this column, I was hoping to present you with a handy-dandy hierarchical tree-structured graphic that illustrated the relationship between all of the different types of electric motors, starting (perhaps) by splitting things up into AC motors and DC motors and branching out from there. Since that time, I've grown to be an older, sadder, and wiser person. I've also come to believe that this is a task beyond the ken of mortal man. Suffice it to say that, if you feel daring, perform a Google search on 'Different types of electric motors' (you will soon wish you hadn't).

One reference comes at things from a different angle. In his book *Motors for Makers: A Guide to Steppers, Servos, and Other Electrical Machines* (**https://amzn.to/3N5ucoZ**), Matthew Scarpino kicks things off with a nice and easy-to-comprehend Motor Selection Flowchart. This starts with a decision symbol that asks: 'Do you need to control/measure the precise angle?' There are two options: 'Yes' and 'No.' If you select 'Yes,' you are directed into an action symbol that succinctly says, 'Choose a stepper or servo motor' (if you select 'No,' you head off into a cascade of options and decisions).

To be fair to Matthew, he does note that his diagram is of use only for making an initial assessment and it doesn't cover all the possibilities. For example, universal motors can operate on both DC and AC power, and if any motor is connected to an encoder or position sensor, then its angle can be measured and controlled. Still and all, this chart does provide a useful starting point.

Stepper and servo motors

When it comes to position control, the two main options are stepper motors ('steppers' for short) and servo motors ('servos' for short).

A stepper motor divides a full rotation into a number of equal steps. Such a motor employs a simple form of open-loop control system because it can be commanded (via a simple sequence of discrete pulses) to move through a precise angle and hold at the designated step without any position sensor or feedback. Steppers find myriad uses in things like analogue clocks, 2D and 3D printers, laser cutters, and robots.

A servo motor is a rotary (or linear) actuator that allows for precise control of angular or linear position, velocity and acceleration. In addition to the motor, a servo includes a sensor to determine its current position and a relatively sophisticated controller to provide a closed-loop control system. My chum Rick Curl recently sent me a link to a YouTube video showing some rather tasty dual-axis servos used to implement a robotic drummer called Zenbot (**https://bit.ly/3xELy6w**).

For the purposes of these discussions, we are focusing on small hobby servos based on DC motors in which everything is presented in a single small package. However, we should note that servos come in all shapes and sizes, including whopping industrial servos (both DC and AC) which may employ external sensors and control modules.

First experiments

In my previous column (*PE*, July 2022), Steve shared some interesting information with respect to the servos we ended up using on our animatronic head, such as the two metal-geared 9g Turnigy TGY-50090 servos of (**https://bit.ly/37pEFNb**) we use to control each eye (Fig.4).

On the off chance you wanted to learn something unexpected, the word 'fossick' is Australian for 'rummage.' Telling you this means I am now in a position to say that I just had a fossick in my treasure chest of bits and pulled out a Tower Pro SG92R Micro Servo (Fig.5), like the ones available from Adafruit (**https://bit.ly/3n0ElbY**).

This little scamp can rotate approximately 180° (90° in each direction). Not shown here is the servo horn, which is a short arm (or pair of arms at 180° to each other) that clip onto the gear wheel sticking out of the top of the servo.

These servos are controlled by means of pulse-width modulation (PWM). In this case, there are two parameters of which we need to be aware: the width of the pulse and the period of the signal (see: Fig.6).

It's the width of the pulse that determines the position of the servo's horn. The 1.5ms shown here will drive the servo to its default (central) position. The period of the pulses (that is, the time between pulses) is less important but, for hobby applications, it's common to use 20ms, which equates to a refresh rate of 50Hz (ie, fifty cycles per second).

Servos may be classed as being analogue or digital. Don't panic, because these appear to be identical from the perspective of the controller, which doesn't know or care and sends out the same pulses regardless. The difference is inside the servo itself. An analogue servo employs analogue circuitry to amplify and process the pulses from the controller and use them to drive the motor to the specified position. By comparison, a digital servo contains a microcontroller, which measures and processes the pulses using digital techniques. Each type has advantages and disadvantages, none of which we need delve into here.

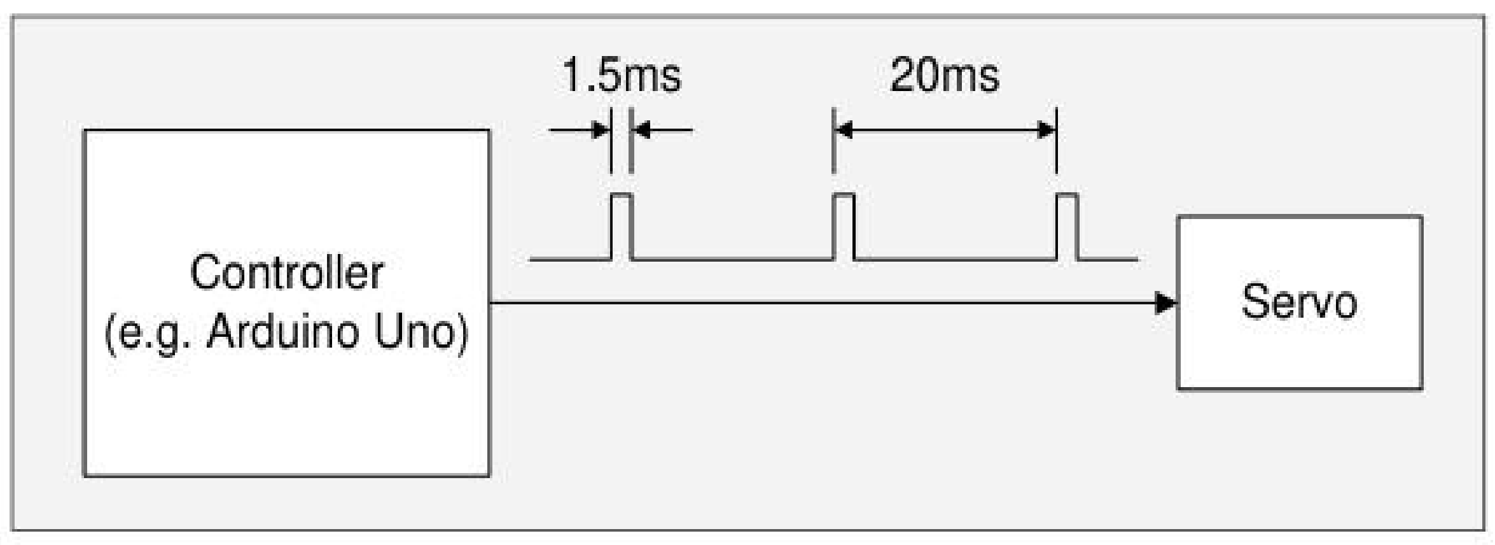


Fig.6. PWM used to control a hobby servo.

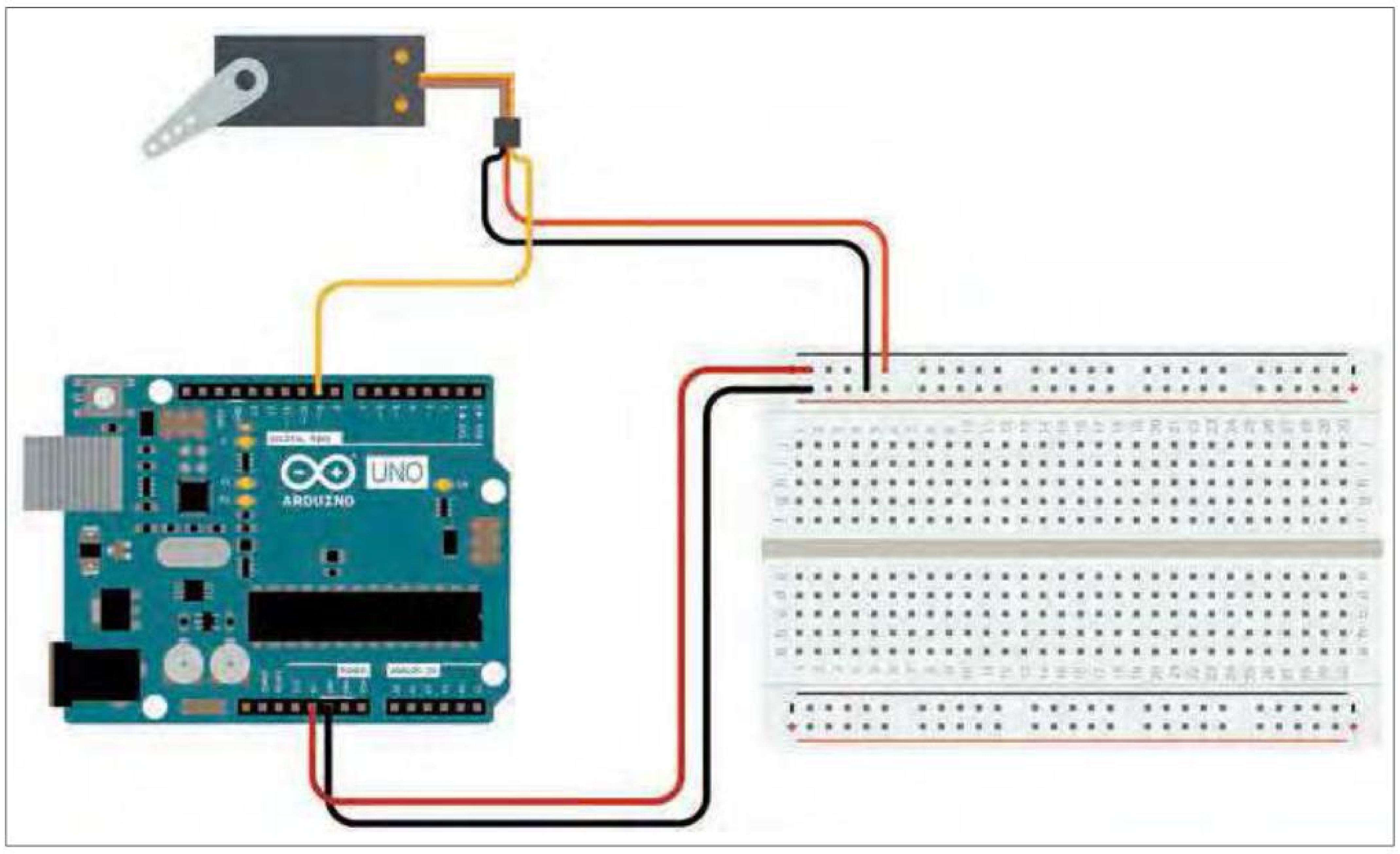
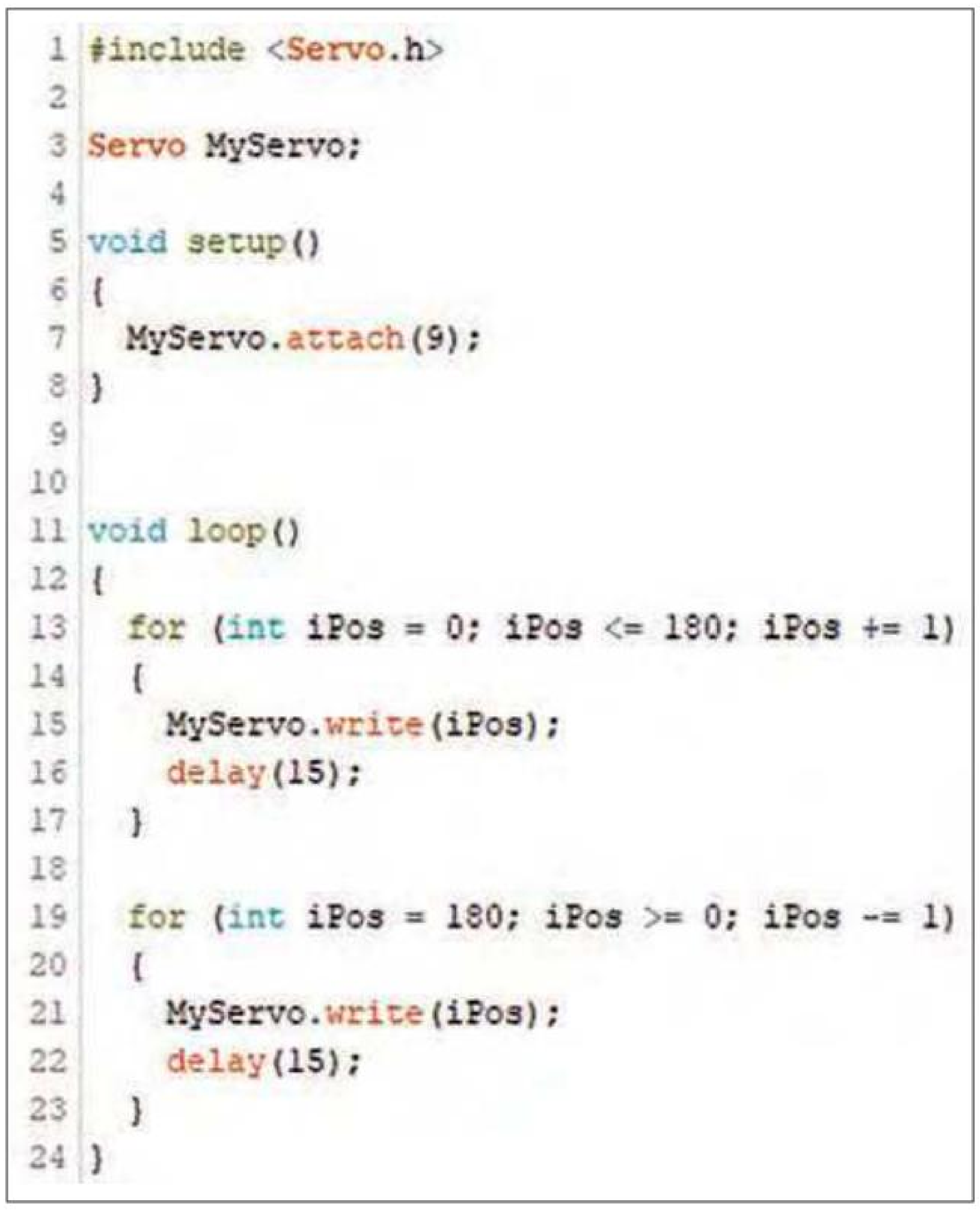


Fig.7. The Sweep Circuit (Image: Adafruit)



```
 1 #include <Servo.h>
 2
 3 Servo MyServo;
 4
 5 void setup()
 6 {
 7   MyServo.attach(9);
 8 }
 9
10
11 void loop()
12 {
13   for (int iPos = 0; iPos <= 180; iPos += 1)
14   {
15     MyServo.write(iPos);
16     delay(15);
17   }
18
19   for (int iPos = 180; iPos >= 0; iPos -= 1)
20   {
21     MyServo.write(iPos);
22     delay(15);
23   }
24 }
```

Fig.8. Example Sweep Circuit program.

Different servos respond to different PWM parameters, although the 1.5ms is always the default position. For example, according to *Motors for Makers*, some servos will accept pulses from 0.7ms (full rotation left/anticlockwise) to 2.3ms (full rotation right/clockwise) (by 'full' we mean the maximum rotation supported by that particular servo). By comparison, according to Adafruit's webpage, in the case of the Tower Pro SG92R, a 1.0ms pulse width corresponds to a rotation of –90°, a 1.5ms pulse corresponds to 0°, and a 2.0ms pulse corresponds to a +90° rotation.

Just to keep us on our toes, different servo manufacturers use different colour schemes for their wires. Common combinations are brown/red/orange, black/red/yellow, black/red/white, and black/red/blue. Fret not! – this is easier than it seems. The red wire, which always appears in the center, is always +ve (let's say 5V, although servos using other voltages are available); the black or brown wires are always –ve (let's say 0V), and the remaining wire is always the control signal wire, irrespective of whether its colour is orange, yellow, white, or blue.

As you will see, controlling our example servo is easy-peasy, not least that a Servo library is included with the Arduino's integrated development environment (**https://bit.ly/3O3rIZu**). The Adafruit website provides some nice experimental circuits and associated programs. For example, consider what they call the 'Sweep Circuit' (Fig.7).

There's no real need to use the breadboard in this case, but doing so will make it easier to migrate to more interesting test cases later. Also, if you happen to have one lying around, it would be a good idea to connect a reasonably large electrolytic capacitor (say 470µF to 1,000µF) between the power and ground rails, thereby helping to insulate the Arduino from any servo-induced power surges, and having the breadboard makes this easy.

The control signal to the servo can use any of the Arduino Uno's digital outputs that supports PWM, which would be pins 3, 5, 6, 9, 10, and 11. We're using pin 9 in this example.

An example sketch (based on the one offered by Adafruit) is shown in Fig.8. Our first step on Line 1 is to include the servo library. On Line 3, we create a servo object. In this case, we've called it `MyServo`, but any legitimate (non-keyword) name will do.

On Line 7 we use the `attach(pin)` method to attach our servo variable to a pin (digital pin 9 in this example). Note that there's another way to do this, which is to use `attach(pin, min, max)`, where `min` defines the pulse width, specified in microseconds, corresponding to the minimum (0 degree) angle on the servo, and `max` defines the pulse width, specified in microseconds, corresponding to the maximum (180°) angle on the servo.

Inside the `loop()` function we have two `for()` loops. The first sweeps the servo from 0° to 180°, then the second sweeps it back from 180° to 0°. The 15ms delays between each step are there to give the servo time to respond and move to its new position. Both of these loops use the `write()` method to specify the desired angle in degrees from 0° to 180° (in the case of a continuous rotation type of servo, writing a value of 0 would set the servo to rotating at full speed in one direction, a value of 180 would set it to rotating at full speed in the other direction, and a value of 90 would result in no movement at all).

Make sure you attach a horn to the servo before you set this program running or you might not even realise that anything is happening. If you've never used a servo before, then seeing the simple back and forth sweep motion presented by this program will bring a little smile to your face because it will make you realise that you've just flung open the door to a world of motion-control possibilities.

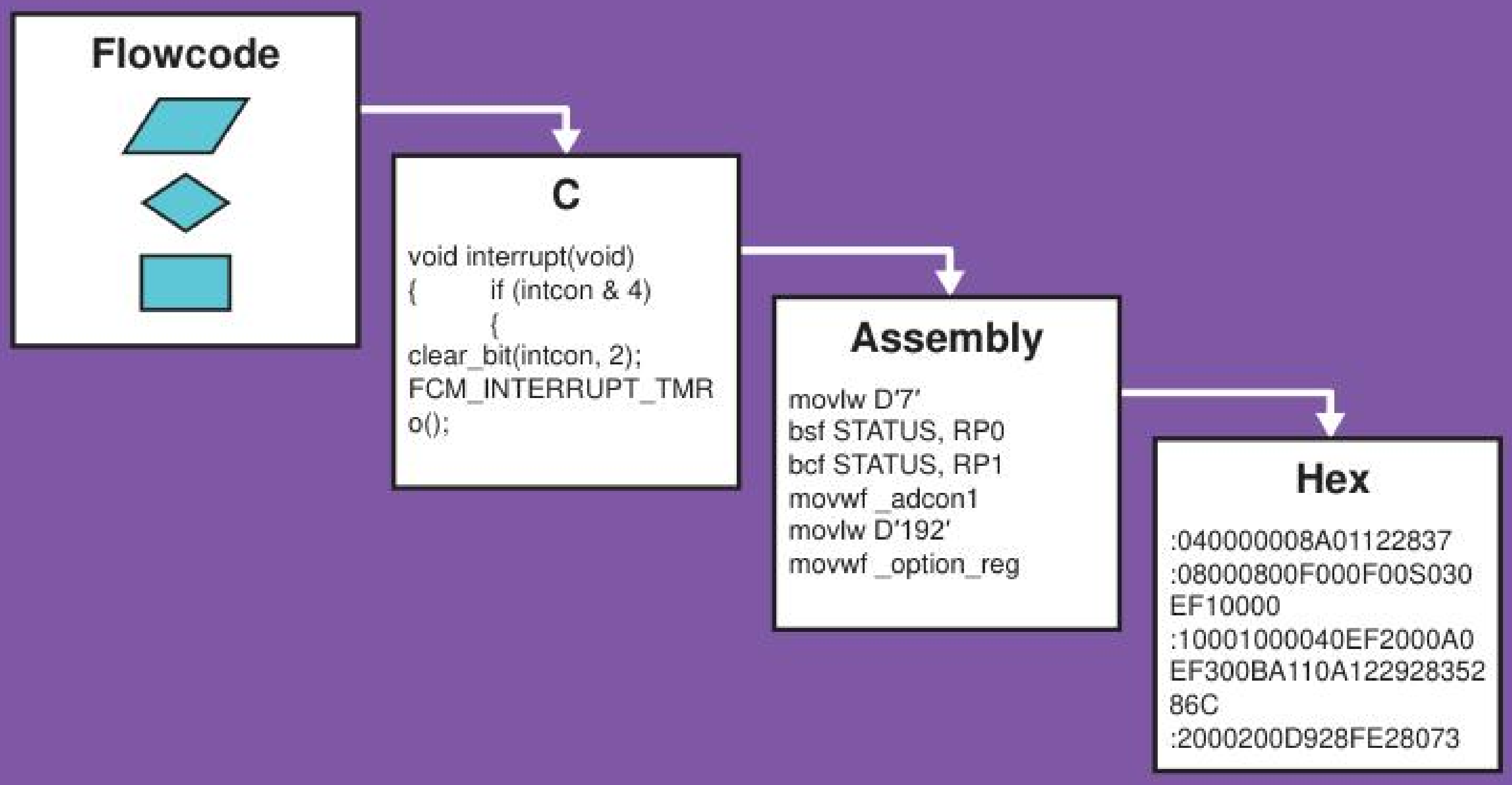
Next time

As exciting as all this is (assuming all this is new to you), we've really dipped only the tips of our toes into the servo control waters. Next time, we're going to consider some different approaches to control a bunch of these little rascals, like the seven servos we're using on our animatronic noggin. Until then, as always, I welcome your questions, comments, and suggestions.

Cool bean Max Maxfield (Hawaiian shirt, on the right) is emperor of all he surveys at **CliveMaxfield.com** – the go-to site for the latest and greatest in technological geekdom.

Comments or questions? Email Max at: **max@CliveMaxfield.com**

Flowcode Graphical Programming



```
void interrupt(void)
{         if (intcon & 4)
          {
clear_bit(intcon, 2);
FCM_INTERRUPT_TMR
o();
```

```
movlw D'7'
bsf STATUS, RP0
bcf STATUS, RP1
movwf _adcon1
movlw D'192'
movwf _option_reg
```

```
:040000008A01122837
:08000800F000F00S030
EF10000
:10001000040EF2000A0
EF300BA110A122928352
86C
:2000200D928FE28073
```

Programming with Flowcode – Part 5: Building a Digital Clock

In this second and final part of the basic RTC Clock series we will cover the User Macros, variables, component functions and some new concept not covered in any previous articles.

The core aim here is to build on last month's description of the algorithm and create a working application in Flowcode. To do this, we will create three User Macros. We have split the program into three parts for two reasons. First, in the free version of Flowcode there is a limit of 20 icons that can be used in any one User Macro and with this constraint our program is too large to fit into one User Macro. Second, it is good programming practice to build program sections out of standalone User Macros. It makes program writing, editing and understanding easier and more efficient. (You can download the full program from the August 2022 page of the *PE* website.)

Variables – global and local

We discussed variable types in some detail last month, so now we need to define and describe the local and global variables that will be used in the program. First, here are the global variables used in the program.

`ReadKeypadValue1`
Byte, Initial value = 0
`ReadKeypadValue1` is used within the main User Macro to read the keypad value. If a key is pressed then the `ChangeSettings` User Macro is called. After the `ChangeSettings` User Macro is exited, then the `ReadKeypadValue1` is used to determine if Hours (`ReadKeypadValue1 = 1`), Minutes (`ReadKeypadValue1 = 2`) or Seconds (`ReadKeypadValue1 = 3`) are adjusted.

`SetTime`
Byte, Initial value = 0
`SetTime` contains the value of either Hours, Minutes or Seconds assigned within the `TimeSetValidity` User Macro. `SetTime` is then used to set the time of the RTC.

`SetTimeStatus`
Byte, Initial value = 0
`SetTimeStatus` is only used within the `TimeSetValidity` User Macro. It is used to count how many times the keypad has been pressed to enter Hours, Minutes or Seconds. For example, if key 1 is pressed, it will allow you to enter one more value and an additional key, eg, `#` or `*`.

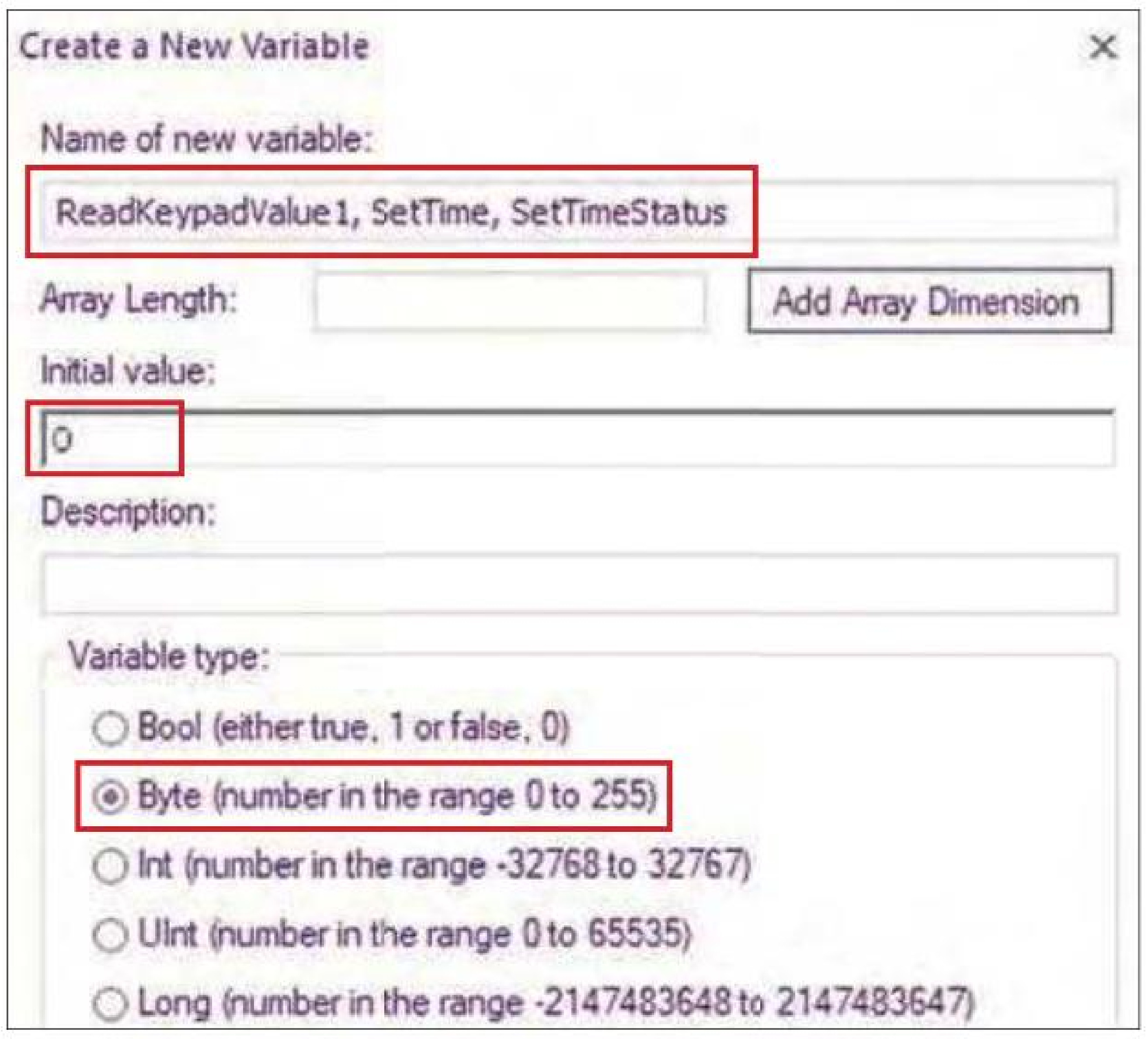
`TempString[2]`
String length of 2 bytes
`TempString` holds the entered time that's being adjusted and converted from the `SetTime` variable, and then displayed on the LCD. A string is used to display the time, rather than a byte because if the time is 09 (in the morning) then a byte will result in only '9' being displayed because all leading zeros are discarded. However, with `TempString`, '09' will be sent to the display.

`TimeString[8]`
String length of 8 bytes
`TimeString` is only used in the `Main` User Macro. It retrieves the time from the RTC; eg, 09:24:58 (nine hours, twenty-four minutes and fifty-eight seconds in the morning), and it is then sent to the display.

Adding the global variables

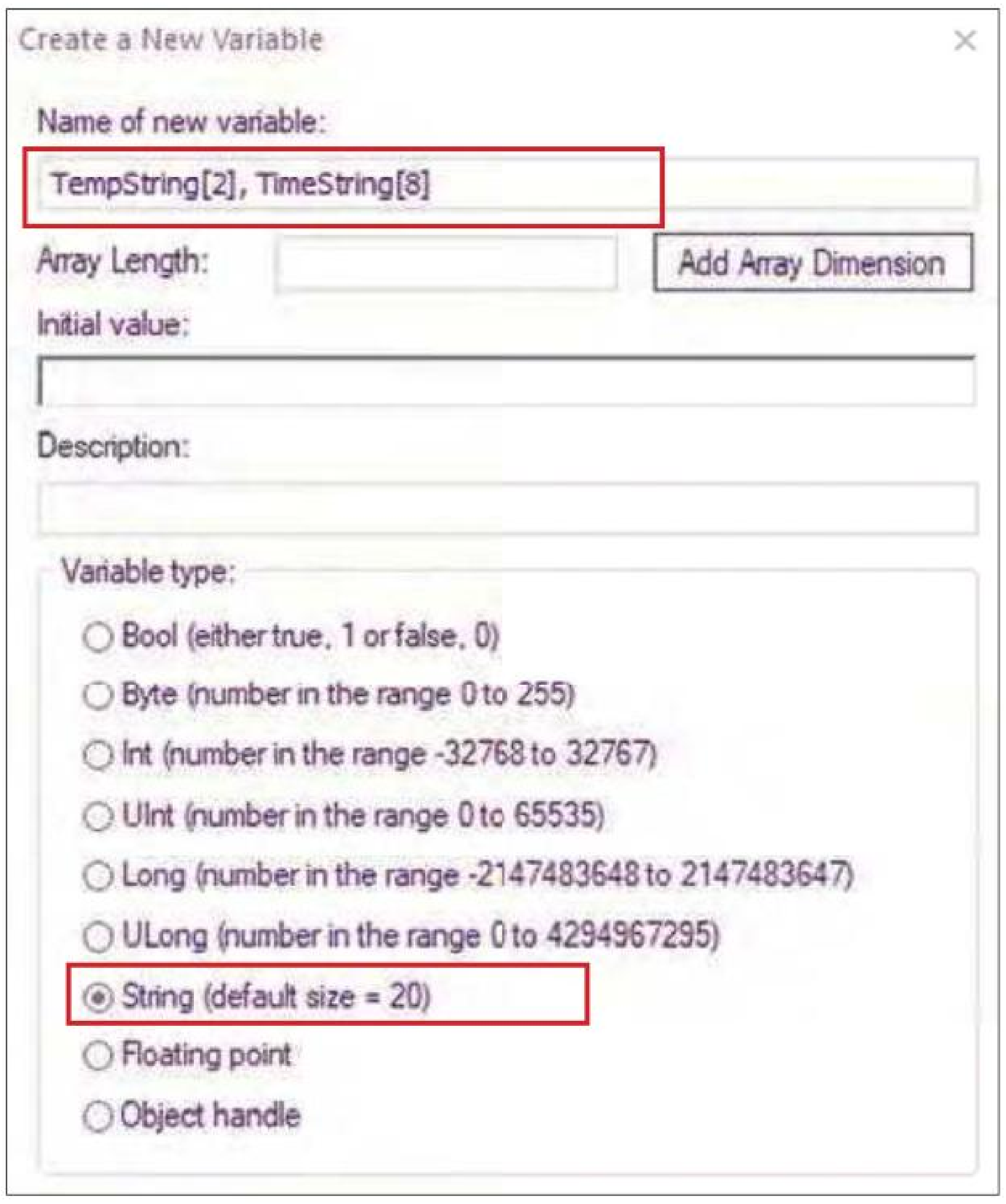
Now we add the global variables. Continuing from last month, open the project to which you have added the project components. (Note: to make the variables easier to follow, I have decided to rename the one shown in *PE*, July page 50 Fig.6. From `KeyPadValue` to `ReadKeypadValue1`.)

Select the Globals icon from project explorer. Notice that the first three global variables listed above are the same type – ie, a byte, initialised with a value of 0. This means we can add all three at the same time, which makes the process quicker, using fewer steps. To do this, use Create a New Variable and enter (including the commas) the names: `ReadKeypadValue1`, `SetTime`, `SetTimeStatus`. Enter 0 for Initial Value and leave Variable type to Byte, as shown in Fig.1. Finally, select OK.



Name of new variable: ReadKeypadValue1, SetTime, SetTimeStatus
Array Length:
Initial value: 0
Description:
Variable type:
○ Bool (either true, 1 or false, 0)
● Byte (number in the range 0 to 255)
○ Int (number in the range -32768 to 32767)
○ UInt (number in the range 0 to 65535)
○ Long (number in the range -2147483648 to 2147483647)

Fig.1. Adding three variables at once with the same variable type and initial value.



Name of new variable: TempString[2], TimeString[8]
Array Length:
Initial value:
Description:
Variable type:
○ Bool (either true, 1 or false, 0)
○ Byte (number in the range 0 to 255)
○ Int (number in the range -32768 to 32767)
○ UInt (number in the range 0 to 65535)
○ Long (number in the range -2147483648 to 2147483647)
○ ULong (number in the range 0 to 4294967295)
● String (default size = 20)
○ Floating point
○ Object handle

Fig.2. Adding multiple string variables.

Multiple string variables can also be added in one hit, but if their lengths are different just add `[variable Length]` at the end of each variable name. For example, since `TempString` has a value of two bytes and `TimeString` has a value of 8 bytes. You just enter: `TempString[2]`, `TimeString[8]`. Remember to change Variable type to String, as shown in Fig.2. Again, select OK.

Local variables

Now we turn to the Local variables. You will recall from last month that they can only be set in a User Macro that uses them. You add them in the Project explorer. Select the Macros Icon, click on the '+' of the appropriate User Macro, which will then show a Variables heading, under which you can double-click on 'Add new'.

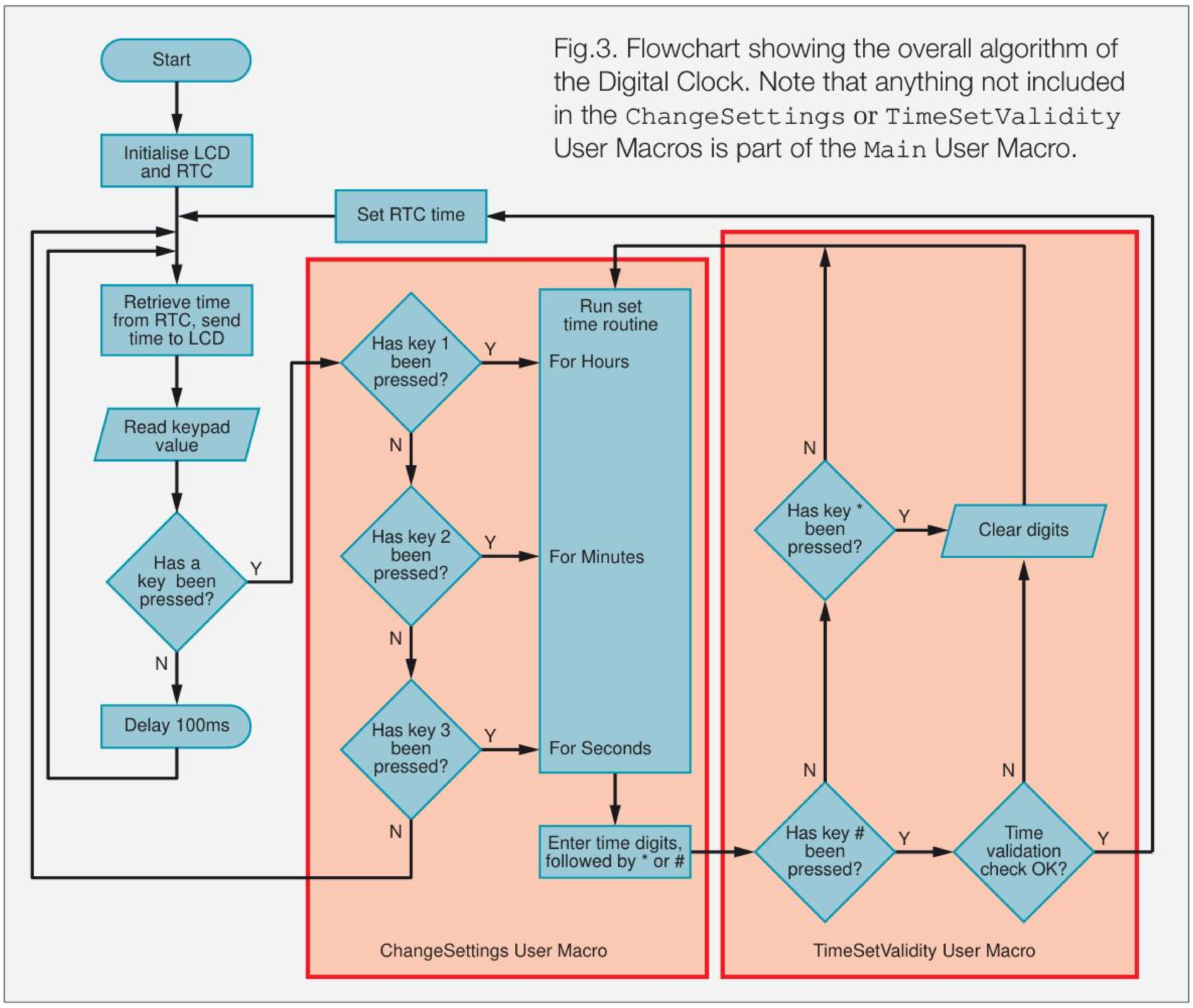


Fig.3. Flowchart showing the overall algorithm of the Digital Clock. Note that anything not included in the `ChangeSettings` or `TimeSetValidity` User Macros is part of the `Main` User Macro.

There are two local variables in the `ChangeSettings` User Macro.

`.ExitChangeSettings`
Bool, Initial value = 0
`.ExitChangeSettings` is used to keep the program running within the loop. If the value is 0 then the loop will keep running until the value of `.ExitChangeSettings` is 1.

`.KeypadValue2`
Byte, Initial value = 0
`.KeypadValue2` is the keypad value when any key is pressed. If no keys are pressed, then the value stays at 255. If the 1 key is pressed then the value of `.KeypadValue2` is 1; if the 7 key is pressed then the value of `.KeypadValue2` is 7, and so on.

The `TimeSetValidity` User Macro has just one local variable.

`.EndTimeSetRoutine`
Bool, Initial value = 0
At the end of the `TimeSetValidity` User Macro there is an expression within a calculation icon: `.Return = .EndTimeSetRoutine`. The value of `.EndTimeSetRoutine` variable is assigned to `.Return`, so when the `TimeSetValidity` User Macro has exited, the value of `.Return` is passed back to the `ChangeSettings` User Macro. We explained last month that this is the only way a local variable can be passed from one User Macro to another. Note that unlike other local variables, you don't create `.Return`, this is done automatically.

`ReadKeypadFromChangeSettings`
Byte, parameter so no initial value
`.KeypadValue2` is passed to this parameter so that it can be used by the `TimeSetValidity` User Macro. See section with Fig.7 in last month's article for an explanation of parameters.

The User Macros

Now we turn to the heart of the program – the three User Macros. Before proceeding it would be a good idea to refresh your memory of the program overview from the end of last month's article.

The flowchart in Fig.3 is an update of Fig.9 from last month, but with two red boxes added. One indicates the location of the `ChangeSettings` User Macro and the other is the `TimeSetValidity` User Macro. The rest of the diagram is the `Main` User Macro.

You can see the User Macros in Fig.4 – this image is just to give you a rough overview of their construction. They are too big to reproduce properly here, so download them from the August 2022 page of the *PE* website. Notice that on these diagrams, each icon is numbered, and we will provide a commentary on each icon's performance and contribution to the program.

The three User Macros are called `Main`, `ChangeSettings` and `TimeSetValidity`. `Main` starts the program, displays the time and does this continuously providing no key is pressed.

If a key is pressed the program transfers to the `ChangeSettings` User Macro, which first checks to see if the key pressed is 1, 2 or 3. If it isn't one of these then the program returns to `Main`. If 1, 2 or 3 has been pressed then the program allows the user to enter Hours, Minutes or Seconds followed by a * (to clear the entry) or # (to accept the entered value).

If the value is accepted then the program moves to the `TimeSetValidity` User Macro, which checks if a viable value has been entered. If not, then it returns the program to the `ChangeSettings` User Macro. If the value is viable then the program moves back to `Main` and updates the time.

Now that you understand the purpose and operation of each User Macro, we will take each one and explain every icon according to the number label on the downloaded diagrams.

Main User Macro

You should know by now that every program must have a User Macro called 'Main'. After the automatically inserted BEGIN icon, the 16 user-inserted icons perform the following roles.

1. **`lcd_16x2 Start`**
This function initialises the display and must be placed before any other LCD component functions – otherwise, the display will be blank. This is an example of a function with no inputs or outputs.

Fig.4 a) `Main` User Macro.

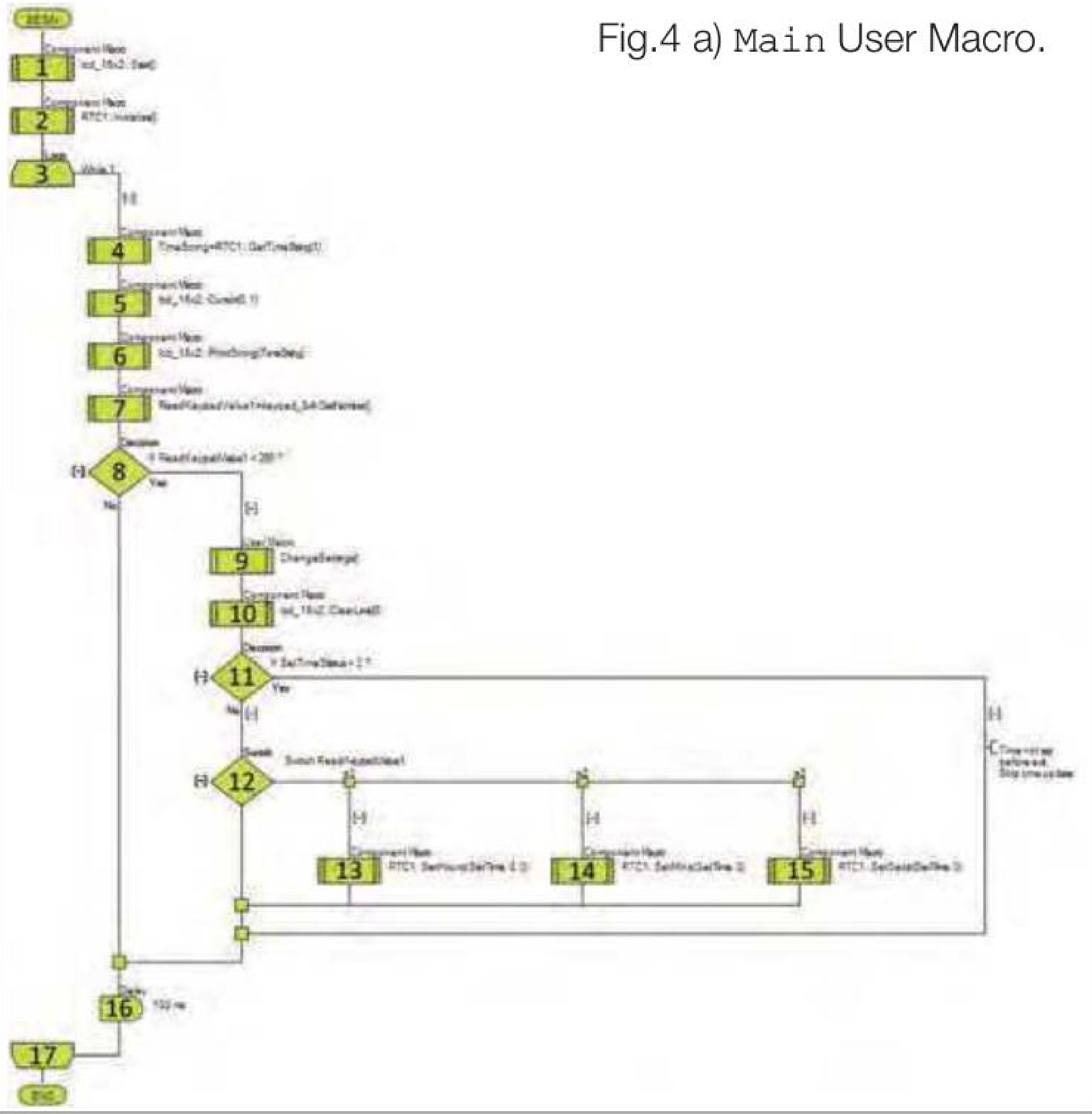


Fig.4. b) `ChangeSettings` User Macro.

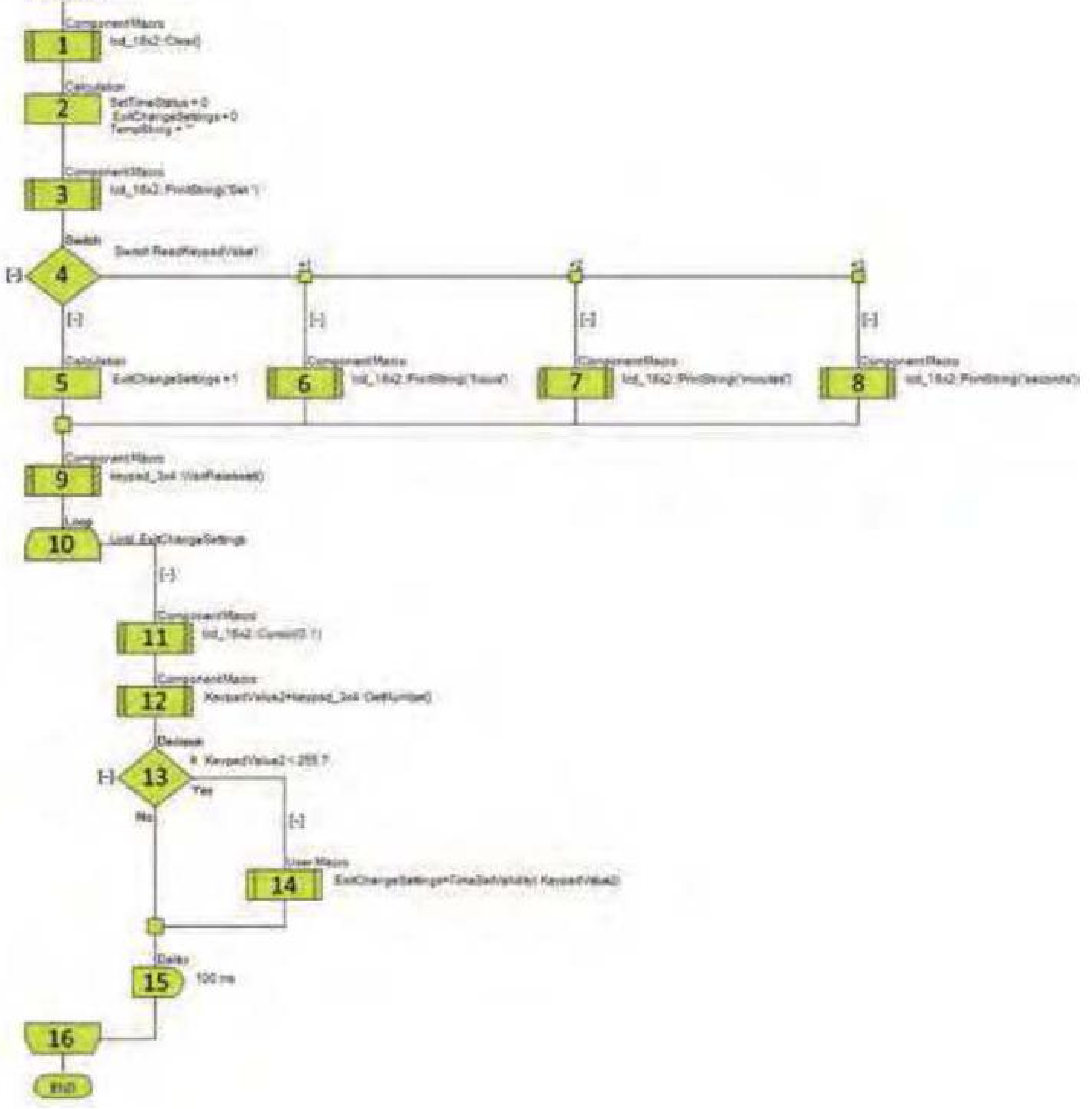


Fig.4. c) `TimeSetValidity` User macro.

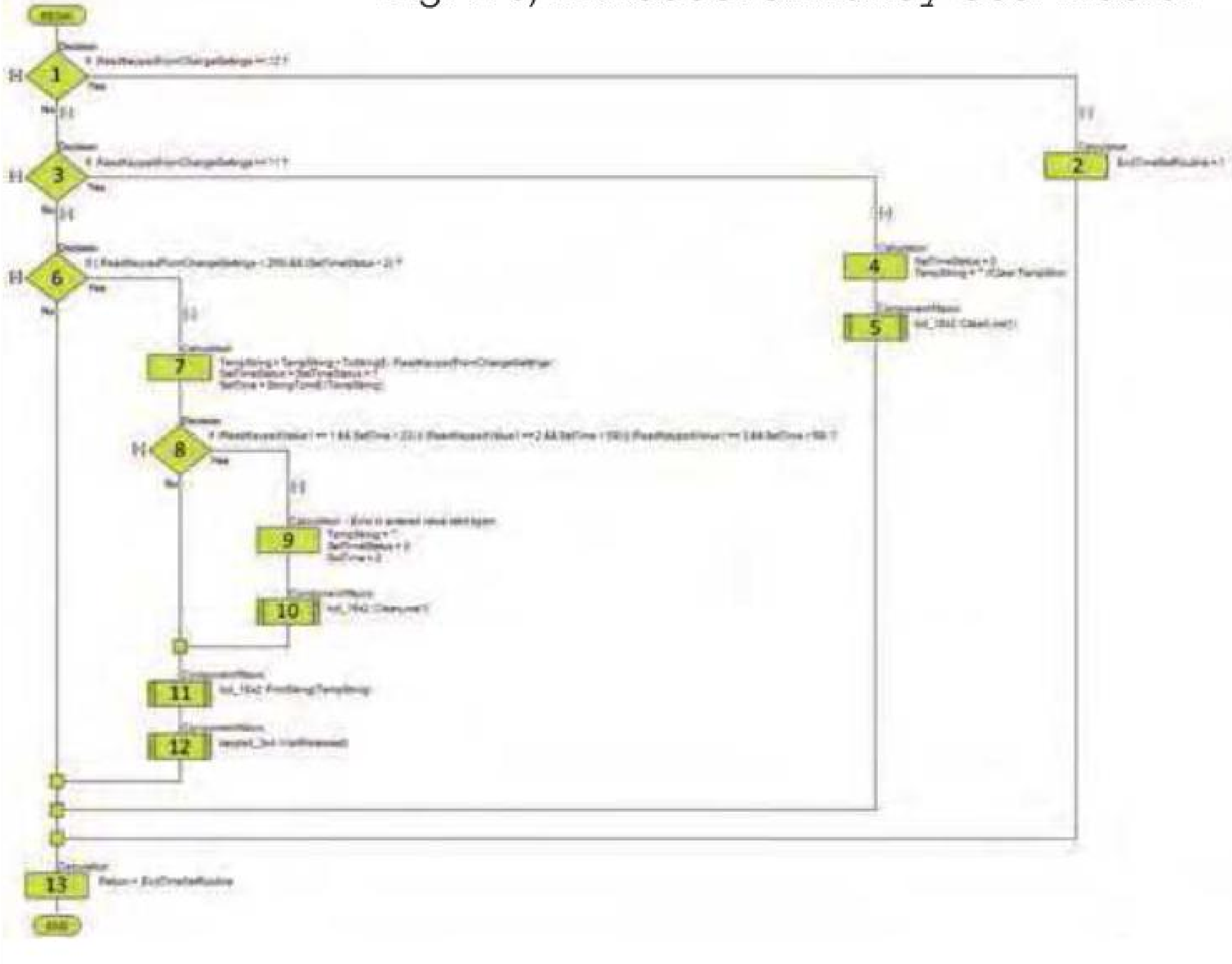


Fig.4. 'Reduced' versions of the three User Macros – download full-size versions from the August 2022 page of the PE website.

2. RTC1 Initialise
This function initialises the RTC and must be placed before any other RTC component functions, otherwise the RTC component will not function.

3. Loop
The loop must be present, so that the code does not reach the end of the User Macro and come to a halt. The loop ensures the code repeats indefinitely.

4. RTC1 GetTimeString, IncludeSeconds = 1, Return Value(STRING) = TimeString
This is a function that requires an input and an output. In the Component properties box, the input is under Expression and the output is the under Return Value, in this case the variable required is a string. If the expression is a 1 then seconds are added to the string; eg, the string variable will be 21:42:34. If the expression is 0 then seconds are not added, and the string variable will be 21:42.

5. lcd_16x2 Cursor, x = 0, y = 1
This function is just an input. The expression values set the cursor position. In this case the cursor on the LCD will be set at the far left (x = 0) and the second row down (y = 1).

6. lcd_16x2 PrintString, Text = TimeString
This is an input, the value of the string (`TimeString`) is sent to the display, starting at the cursor position set in icon 5.

7. Keypad_3x4 GetNumber, Return Value(BYTE) = ReadKeypadValue1
When a keypad button is pressed the value returned to the `ReadKeypadValue1` variable represents the number of the keypad, except for the * and # keys which return 11 and 12, respectively.

8. Command Icon Decision: KeypadValue < 255
This statement will only be true (Yes branch) if a button is pressed and thus the value of `KeypadValue` becomes less than the default value of 255 (no buttons pressed).

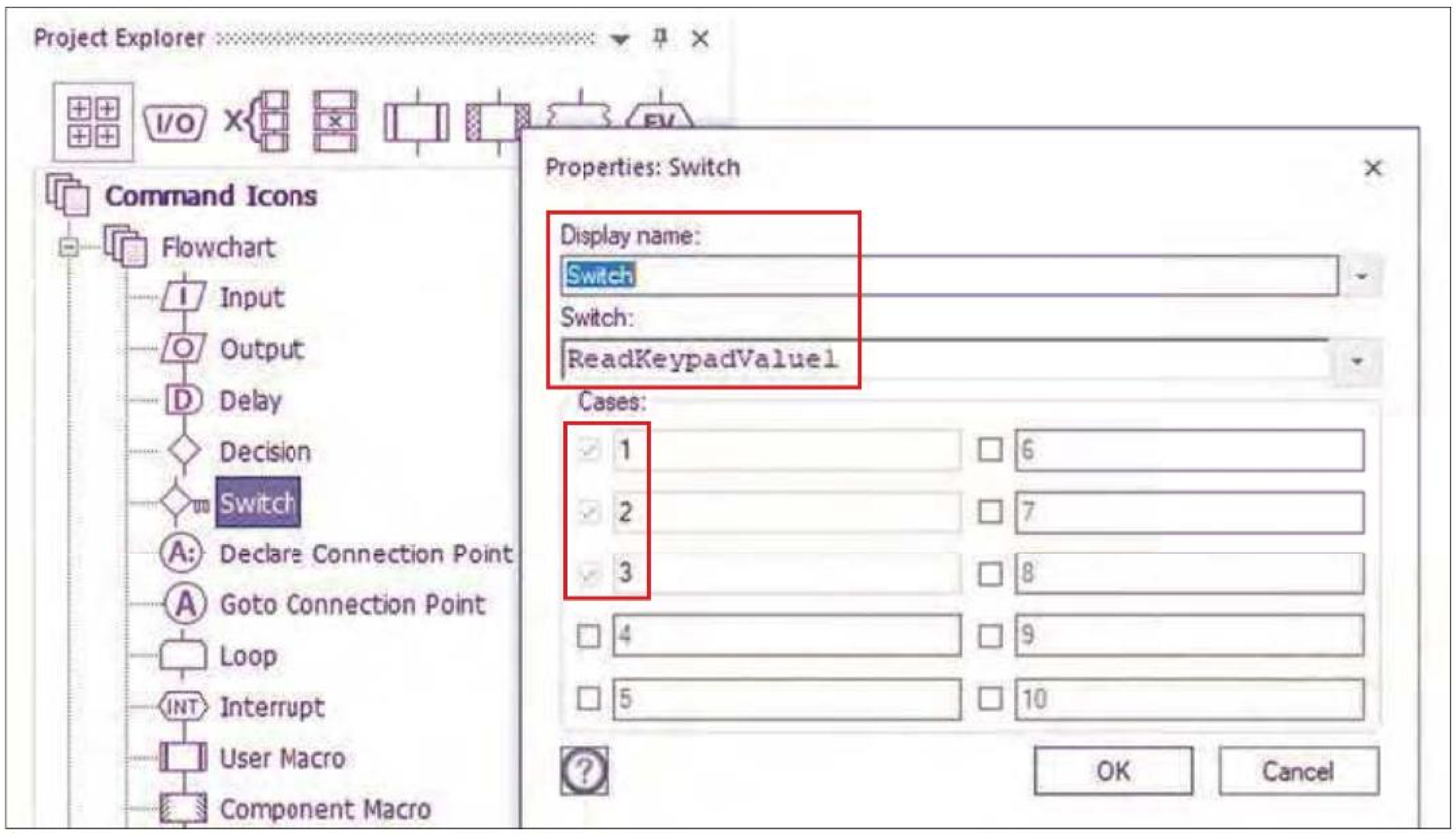
9. User Macro = ChangeSettings
This icon calls the `ChangeSettings` User Macro when it needs to be accessed.

10. lcd_16x2 Clearline, Line = 0
Clears the top row of the display.

11. Command Icon Decision: < 2
The Yes branch will only be accessed if the `SetTimeStatus` is less than 2. If that is the case the RTC will not be updated with the new time.

12. Command Icon Switch, Switch = KeypadValue, Cases =1, 2, 3
Setting up the Command Icon Switch is a little more complicated than other icons. Fig.5. shows the details. The Cases are the values expected. So, if `ReadKeypadValue1 = 1` then the =1 branch will be accessed (Icon 13). If `ReadKeypadValue1 = 2` then the =2 branch will be accessed (Icon 14) and so on. If none of the Cases match the `ReadKeypadValue1` variable, then only the branch below the diamond icon is accessed.

13. RTC1 SetHours, Hours = SetTime, Mode = 0, WhichClock = 0
This sets the hours of the RTC. There are three inputs. Hours is set with the contents of the `SetTime` variable. Mode sets a 24-hour format (`Mode = 0`) or a 12-hour format (`Mode = 1`). For



Display name: Switch
Switch: ReadKeypadValue1
Cases:
☑ 1 ☐ 6
☑ 2 ☐ 7
☑ 3 ☐ 8
☐ 4 ☐ 9
☐ 5 ☐ 10

Fig.5. Setting up the Command Icon Switch.

possible future upgrades to the project, `WhichClock` sets the type of clock: `Main` (`WhichClock = 0`), `Alarm1` (`WhichClock = 1`) or `Alarm 2` (`WhichClock = 2`).

14. RTC1 SetMins, Minutes = SetTime, WhichClock = 0
Sets the minutes of the RTC

15. RTC1 SetSecs, Seconds = SetTime, WhichClock = 0
Sets the seconds of the RTC

16. Command Icon Delay = 100ms
Adds a delay time of 100 milliseconds, so the display is not updating too rapidly.

17. Loop end
You do not need to add anything to this icon; it's the end of the program, so program flow jump back to icon 3.

ChangeSettings User Macro

Next, the `ChangeSettings` User Macro, which is called only when a key on the keypad has been pressed.

1. lcd_16x2 Clear
Clears the whole display; there are no inputs or outputs.

2. Command Icon Calculation: SetTimeStatus = 0
This is just a calculation. When this icon is accessed, `SetTimeStatus` and `.ExitChangeSettings` are assigned to `0` (Note the period is required at the start because `.ExitChangeSettings` is a local variable.) Also, the string variable `TempString` is cleared.

3. lcd_16x2 PrintString, Text = "Set "
Note the deliberate space after `Set` and before the closing double quote mark.

4. Command Icon Switch, Switch = ReadKeypadValue1
There are three cases, 1, 2 and 3. As before (Fig.5) enter `ReadKeypadValue1` for the variable name (below 'Switch:') and only tick cases 1, 2 and 3.

5. Command Icon Calculation: .ExitChangeSettings = 1
This is placed below the diamond icon, so if any value other than 1, 2 or 3 is entered then `.ExitChangeSettings` will change from 0 to 1. That in turn allows the loop to exit from the end of the User Macro at icon 16.

6. lcd_16x2 PrintString, Text = "hours"
This is placed within branch = `1`, so it will only be accessed if 1 is pressed on the keypad.

7. lcd_16x2 PrintString, Text = "minutes"
This is placed within branch = `2`, so it will only be accessed if 2 is pressed on the keypad.

8. lcd_16x2 PrintString, Text = "seconds"
This is placed within branch = `3`, so it will only be accessed if 3 is pressed on the keypad.

9. WaitReleased
The program will halt here if the keypad is detected as being pressed. The program will continue as soon as the keypad has been released.

10. Command Icon Loop, Loop until selected, Expression = .ExitChangeSettings
Test the loop at the start and if `.ExitChangeSettings` remains at 0 then the loop will carry on until `.ExitChangeSettings` changes from 0 to 1

11. lcd_16x2 Cursor, x = 0, y = 1
Set the cursor position starting at 0. Here, the cursor on the LCD will be set at the far left (`x = 0`) and the second row down (`y = 1`).

12. Keypad_3x4 GetNumber, Return Value:(BYTE) = .KeypadValue2
If the keypad is pressed then the key pressed will be assigned to `.KeypadValue2` variable.

13. Command Icon Decision: .KeypadValue2< 255
If the keypad is not pressed, then the value will default to 255, so the No branch is accessed. As soon as a key is pressed then the Yes branch will be accessed as the value will be between 0 and 12.

14. User Macro TimeSetValidity, ReadKeypad = .KeypadValue, Return Value(BYTE) = .ExitChangeSettings
The value of local variable `.KeypadValue2` is passed to the input of the `TimeSetValidity` User Macro. As soon as the `TimeSetValidity` User Macro has ended, the value of `.ExitChangeSettings` in the `TimeSetValidity` User Macro will be passed back to the `ChangeSetting` User Macro.

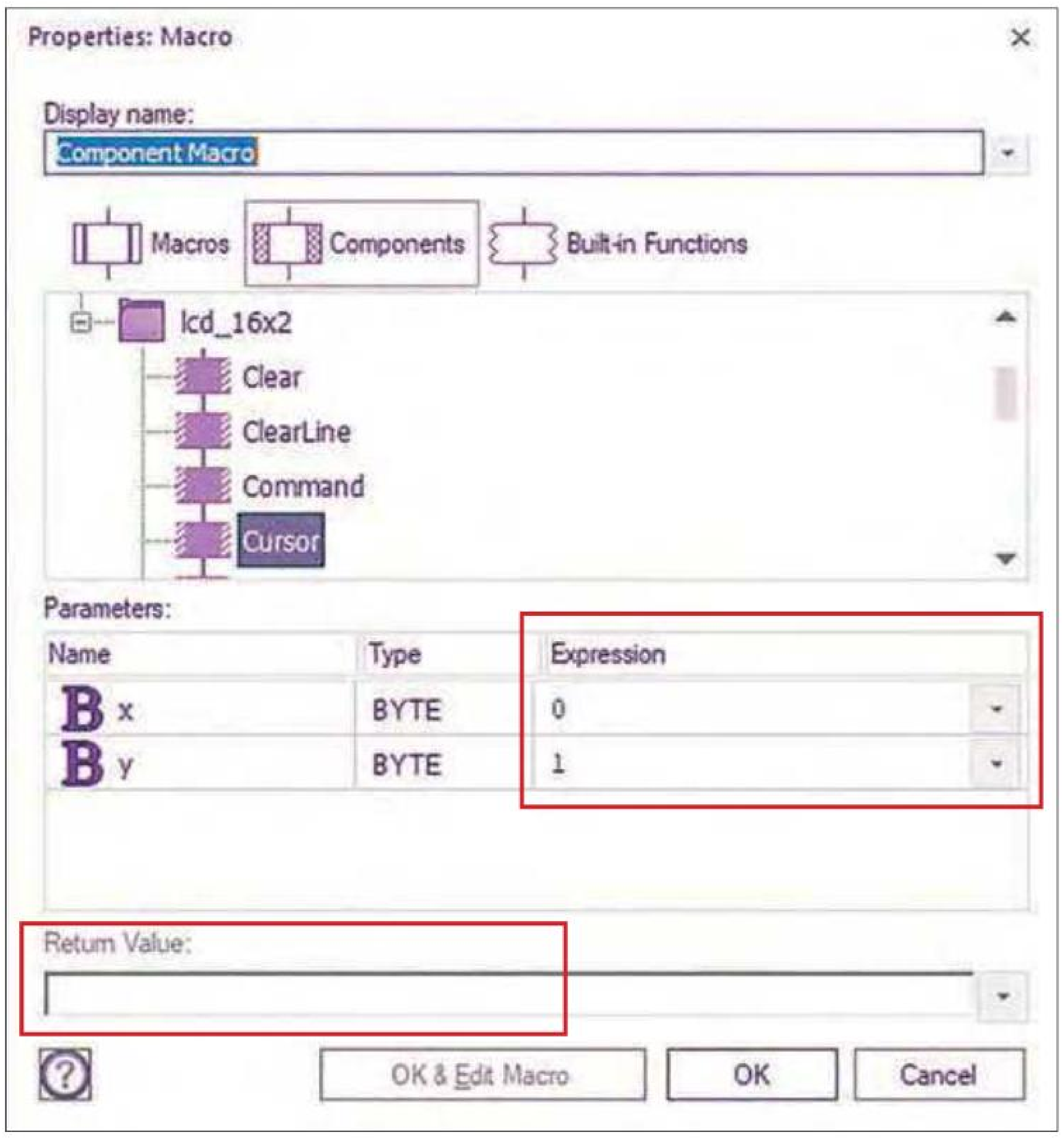
15. Command Icon Delay = 100ms
Adds a delay of 100ms to prevent the display updating too rapidly.

16. Loop end
Only exit loop if `.ExitChangeSettings` has been changed from 0 to 1, otherwise go back to icon 10.

TimeSetValidity User Macro

The third User Macros is `TimeSetValidity`. This is only called after you have a) chosen via the keypad which time parameter to set (hours (1), minutes (2) or seconds (3)) and b) entered the first digit of a value. (See `ChangeSettings` User Macro, icon 14.)

1. Command Icon Decision: .ReadKeypadFromChangeSettings == 12
The Yes branch will be accessed if the # key is pressed (which has a key value of 12).



Display name: Component Macro

Name	Type	Expression
x	BYTE	0
y	BYTE	1

Fig.6. Setting up the `Cursor` function of the `lcd_16x2`.

2. Command Icon Calculation: .EndTimeSetRoutine = 1
`.EndTimeSetRoutine` is assigned 1 only if the Yes branch (1) was accessed, otherwise it will be the default value of 0.

3. Command Icon Decision: .ReadKeypadFromChangeSettings == 11
The Yes branch will be accessed if the `*` key is pressed (which has a key value of 11).

4. Command Icon Calculation: SetTimeStatus = 0
Assigns the `SetTimeStatus` with 0; and `TempString = ""` (the two empty quote marks clear the string).

5. lcd_16x2 ClearLine(1)
Clears the second row of the LCD.

6. Command Icon Decision: (.ReadKeypad < 255) && (SetTimeStatus < 2)
The Yes branch is accessed only when a key is pressed (255 = no key pressed) AND if the `SetTimeStatus` is less than 2.

7. Command Icon Calculation: TempString = TempString + ToString$ (.ReadKeypad)
The keypad is read and the `ToString$` coverts the keypad value from a byte to a string; the result is added to the `TempString` variable. `SetTimeStatus = SetTimeStatus + 1` increments the value of the `SetTimeStatus` by 1; `SetTime = StringToInt$ (TempString)` converts the value of `TempString` back to a byte value and is assigned to the `SetTime` variable.

8. Command Icon Decision: (ReadKeypadValue1 == 1 && SetTime > 23) || (ReadKeypadValue1 == 2 && SetTime > 59) || (ReadKeypadValue1 == 3 && SetTime > 59)
The Yes branch is accessed if (`ReadKeypadValue1` AND `SetTime` is greater than 23) OR if (`ReadKeypadValue1` is 2 AND `SetTime` is greater than 59) OR if (`ReadKeypadValue1` is 3 AND `SetTime` is greater than 59).

9. Command Icon Calculation
`TempString = ""` clears the `TempString` variable. `SetTimeStatus = 0` assigns the `SetTimeStatus` variable with 0. `SetTime = 0` assigns the `SetTime` variable with 0.

10. lcd_16x2 ClearLine(1)
Clears the second row of the display.

11. lcd_16x2 PrintString, Text = TempString
Display the value of the `TempString` variable.

12. keypad_3x4 WaitReleased
Stop at this point if any key of the keypad is pressed.

13. Command Icon Calculation: .Return = .EndTimeSetRoutine
Assign the `.Return` variable with the value of `.EndTimeSetRoutine` variable. (Important point: you don't add the `.Return` variable. It is automatically added when a Return type is selected within the dropdown at the bottom of Create a New Macro pop up.)

Components function types

Component functions (eg, functions used by displays or keypads) can be one of four types in terms of inputs and outputs. It's important to know which is which, so here is a brief overview.

No input or output required
At the start of the `Main` User Macro, examine `lcd_16x2 Start` (icon 1). Any variables within `()` after the name represents the input value, but since the brackets are empty no input is required. (Note that `Start` is the function, and `lcd_16x2` is the component.)

Only an input is required
Next, consider `lcd_16x2 Cursor` in the `Main` User Macro, icon5. Note the two sets of numbers which represents the cursor `x` and `y` position, separated by a comma. These inputs can be a number or a variable which is compatible for the component. All inputs are entered under the Expression column as shown in Fig.6. Note in Fig.6 that the `Return Value` is greyed out because a returned value would be an output, which in this case is not needed.

Only an output is required
If you look at the `Main` User Macro, icon 7, the component function is `keypad_3x4 GetNumber` which occurs when a button is pressed. The output value is returned to the `ReadKeypadValue1` variable.

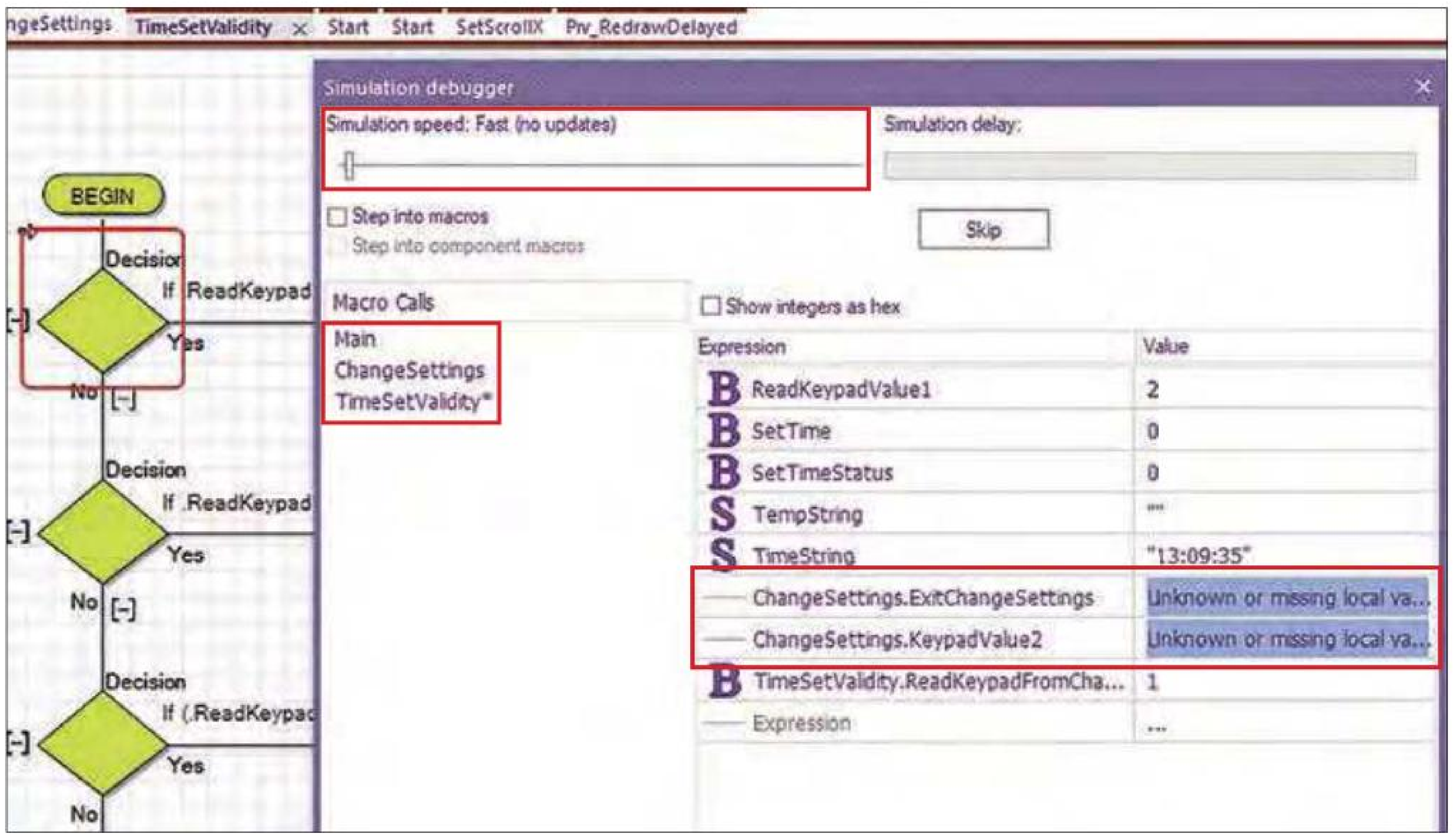
Both input and output are required
Last, look at the `Main` User Macro, icon 4. Here, the input for `RTC1 GetTimeString` is the `1` in the brackets, while the output is returned to the `TimeString` variable.

Finally, note that a component does not necessarily always operate with just one of the above (eg, only input or only output), it depends on which function of the component being used. For example the first two categories above both used the component `lcd_16x2` but with different functions, `Start` and `Cursor`.

Setting up a simulation

Now that we have the three User Macros, it's time to look at using Flowcode's simulation tools. We won't go over the full simulation procedure – we have already covered that (see: *PE*, February 2022), but here are some points worth emphasising.

When running a simulation in Flowcode, the key parameters to watch as we step through or run the program are the variables. Before running a simulation you can check what variables are in the program by selecting the appropriate Globals or Locals icons within Project explorer (if closed, Project explorer can be opened within the view menu). Naturally, local variables, if any, can only be seen if you select the appropriate User Macro tab at the top of each User Macro flowchart. In this project, `Main` doesn't use local variables, so you would need to select either the `ChangeSettings` or `TimeSetVailidity` tab to see local variables.



Simulation speed: Fast (no updates)

Simulation delay:

☐ Step into macros

☐ Step into component macros

☐ Show integers as hex

Macro Calls:
Main
ChangeSettings
TimeSetValidity*

Expression	Value
B ReadKeypadValue1	2
B SetTime	0
B SetTimeStatus	0
S TempString	""
S TimeString	"13:09:35"
ChangeSettings.ExitChangeSettings	Unknown or missing local va...
ChangeSettings.KeypadValue2	Unknown or missing local va...
B TimeSetValidity.ReadKeypadFromCha...	1
Expression	...

Fig.7. The simulation debugger window – note the speed slider and 'Unknown' values.

To start the simulation press the F8 key, which will open the Simulation debugger window. Referring to Fig.7, select the global variables you wish to watch, by clicking on '…' in the Value field for Expression. Then choose Add variable from the resulting pop up and select the Globals or Locals icon, depending on what you are interested in simulating.

Again, remember if to want to add and watch local variables they will only be available for adding if either you or the simulator have accessed the appropriate User Macro tab.

While the simulator is running the current User Macro will be shown on the left of the simulation debugger at the bottom of the list and the User Macro which called it will appear immediately above it; again, see Fig.7.

Also, since the values of local variables can only be viewed when in the appropriate User Macro, if you are in a different User Macro then you will see 'Unknown or missing local variable' highlighted in blue, as shown in Fig.7. This is normal, the correct value will be displayed when the relevant User Macro is accessed.

With local variables, on the simulation debugger, the User Macro name is shown first, then a period/full stop, then the name of the local variable. For example, `.ExitChangeSettings` will be shown as `ChangeSettings.ExitChangeSettings`

Three periods/full stops (...) after a variable means that part of the name is hidden as it is too long for the window width. To adjust the width, hover the cursor between any variable and the value, then the cursor will change to double arrows, hold the left mouse button down and drag the variable window size.

At the top of Fig.7 you can see the Simulation slider. Slide it so that it changes from Fast (no updates) to Fast (with updates). This setting means you can see the variables values change.

Running a simulation

After the variables have been added to the simulation make sure you can view the 2D Dashboard panel (see *PE*, January 2022). We are now ready to run the simulation. Run the program by pressing Go in the Debug ribbon. If all is working, then you should see the time constantly updating in the LCD on the 2D Dashboard panel.

Now test setting the time. Press 1 on the keypad, the display should change from displaying the time to the words 'Set hours', shown on the top line of the display. Try pressing an invalid time eg, 72. You should see the numbers disappear as the program does not allow 72 to be set for hours. Add a valid time (12) and then press the * to clear your entry or press # to enter the value. (Note that there is an important RTC limitation when simulating in Flowcode – the only time which can be displayed is the local PC time, you cannot set the display to a random non-PC time. Of course, on your actual project hardware this limitation will not apply.

Debugging software

When you run a simulation, it is important to note any information shown in the Icon Lists window. This window doesn't have to be always open because it will automatically pop up if the simulation produces an issue that needs highlighting.

For example, with the simulation stopped, right-click on the `Keypad_3x4 Getnumber` component within the `Main` User Macro and select Disable. Now remove the initial value of `ReadKeypadValue1` variable. Run the simulation and the Icon List should pop up showing 'Uninitialized variable' (see Fig.8), indicating an error that needs correcting. (An uninitialized variable is a real problem – it will usually cause your hardware to contain a random value, resulting in unexpected performance.)

Setting up the hardware

In this project, we've focused the bulk of both articles on the software, but what about the hardware?

The hardware should be built on breadboard. It's a straightforward build – just follow the schematic in Fig.9. If you built the LCD Temperature Monitor (*PE*, April 2022) then you should recognise the wiring for the display. The rest of the connections to the keypad and RTC are not difficult but do pay attention to the resistors – there are two values and these are wired differently.

When the wiring is finished you are ready to send the code to the target device. Now power it up. Once the hardware is running the LCD will show the time, but it will be a random time because you haven't set up the RTC with the correct time. Use the keypad to enter the correct Hours, Minutes and Second. Fig.10 shows the system up and running.

Fault finding hardware

Even the simplest circuits can exhibit faults. The following are the most common problems you are likely to encounter.

Display is showing one row of rectangles

First check you have initialised the display in the `Main` User Macro. If that has been done correctly then check the connections to the display's data, RS and RW lines. Make sure they match the schematic (Fig.9). Also, check that the connections to the LCD in its component properties in Flowcode match the hardware in the schematic. Do note that RW is not shown in component properties since it is just connected to GND.

Display is blank

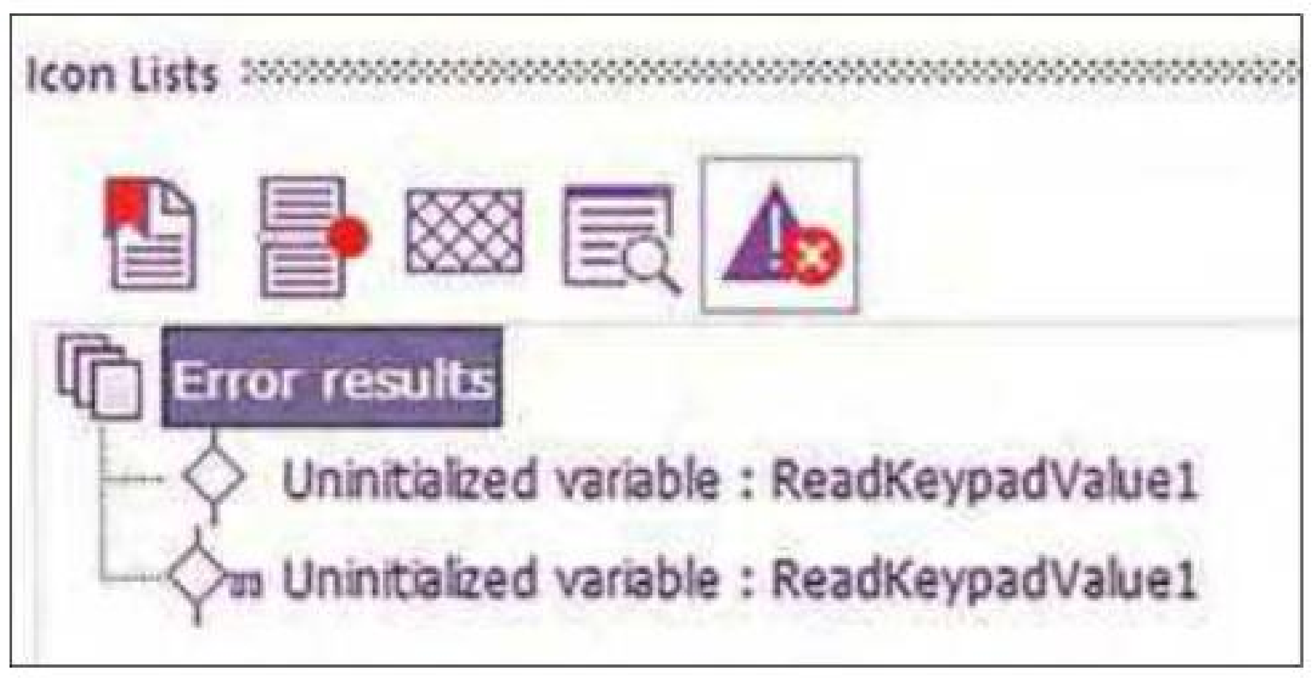
Check the supply pins are at the correct voltage and vary the contrast trimmer. Verify the voltage on the contrast pin (3) varies from 0 to 5V.

Display is showing two rows of rectangles

This normally means the contrast voltage is wrong. Follow the 'display is blank' fault finding.

Display shows a random time

For example, the display shows 02:01:00 and it does not change. This will be an issue between the RTC and the RedBoard/Uno. Check the connections match for Flowcode and your hardware. The SDA and SCK connections could be reversed on your hardware, or open circuit. Always check the correct power supply is going to the RTC pins.



Uninitialized variable : ReadKeypadValue1
Uninitialized variable : ReadKeypadValue1

Fig.8. Icon Lists window highlighting an error during a simulation.

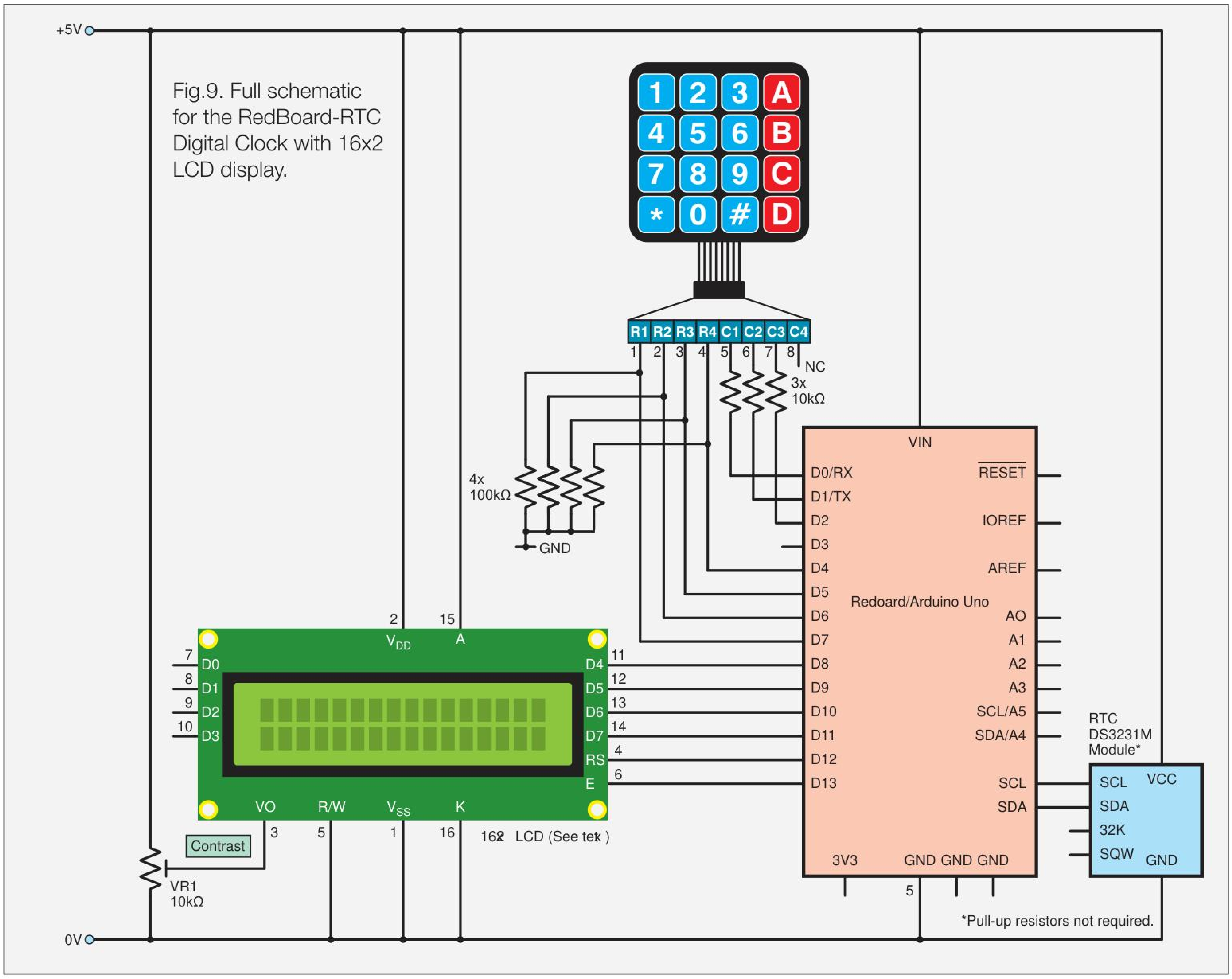

+5V
0V
1 2 3 A
4 5 6 B
7 8 9 C
* 0 # D
R1 R2 R3 R4 C1 C2 C3 C4
1 2 3 4 5 6 7 8
NC
3x 10kΩ
4x 100kΩ
GND
VIN
D0/RX
D1/TX
D2
D3
D4
D5
D6
D7
D8
D9
D10
D11
D12
D13
Redoard/Arduino Uno
RESET
IOREF
AREF
AO
A1
A2
A3
SCL/A5
SDA/A4
SCL
SDA
3V3
GND GND GND
5
RTC DS3231M Module*
SCL
VCC
SDA
32K
SQW
GND
2
15
V_{DD}
A
7 D0
8 D1
9 D2
10 D3
11 D4
12 D5
13 D6
14 D7
4 RS
6 E
VO
R/W
V_{SS}
K
3
5
1
16
16x2 LCD (See text)
Contrast
VR1 10kΩ

Fig.9. Full schematic for the RedBoard-RTC Digital Clock with 16x2 LCD display.

*Pull-up resistors not required.

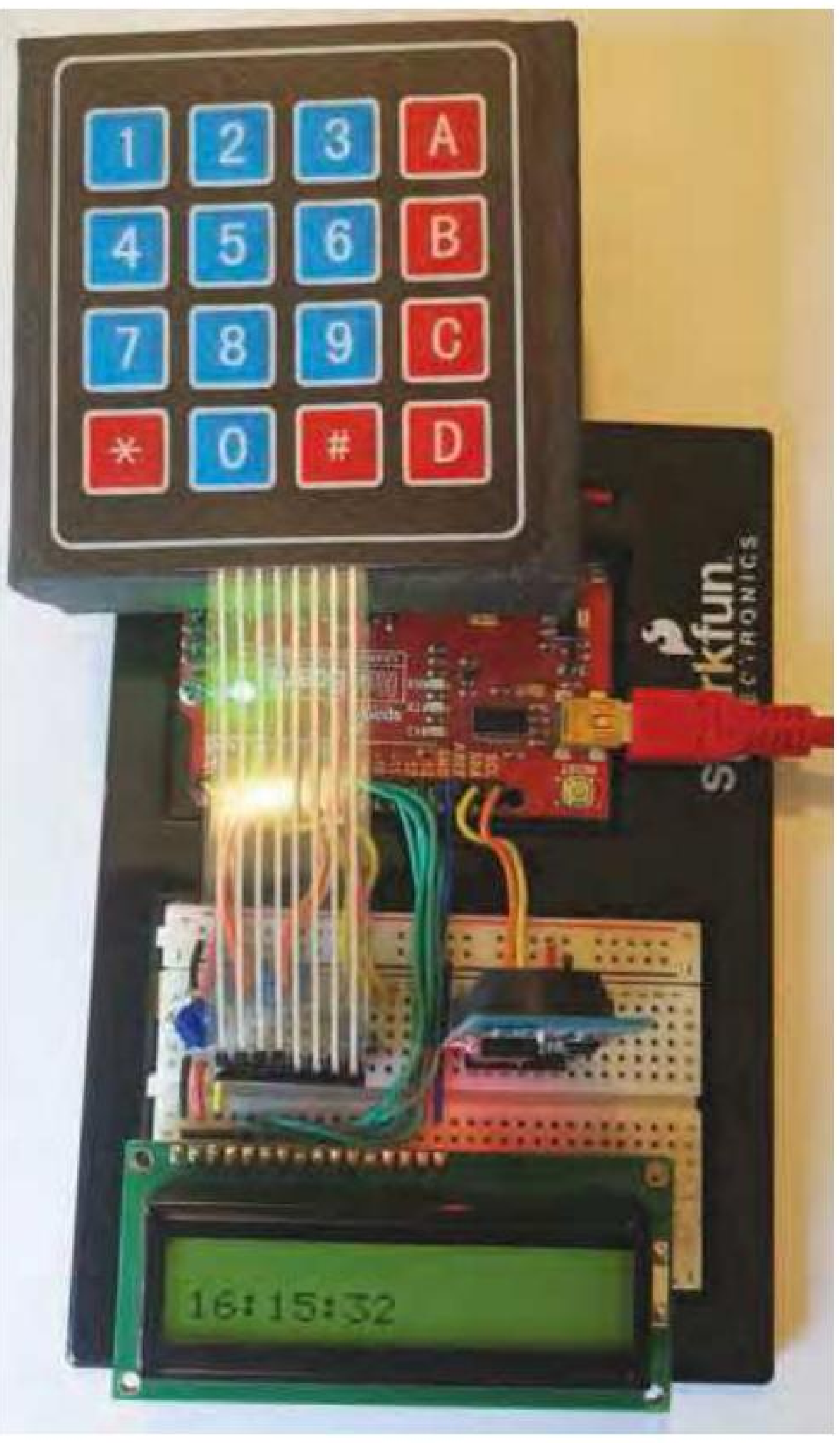

16:15:32
sparkfun

Fig.10. The Digital Clock up and running.

Martin Whitlock is an Applications Engineer at Matrix TSL – which is the company behind Flowcode.

Try Flowcode for *free!*

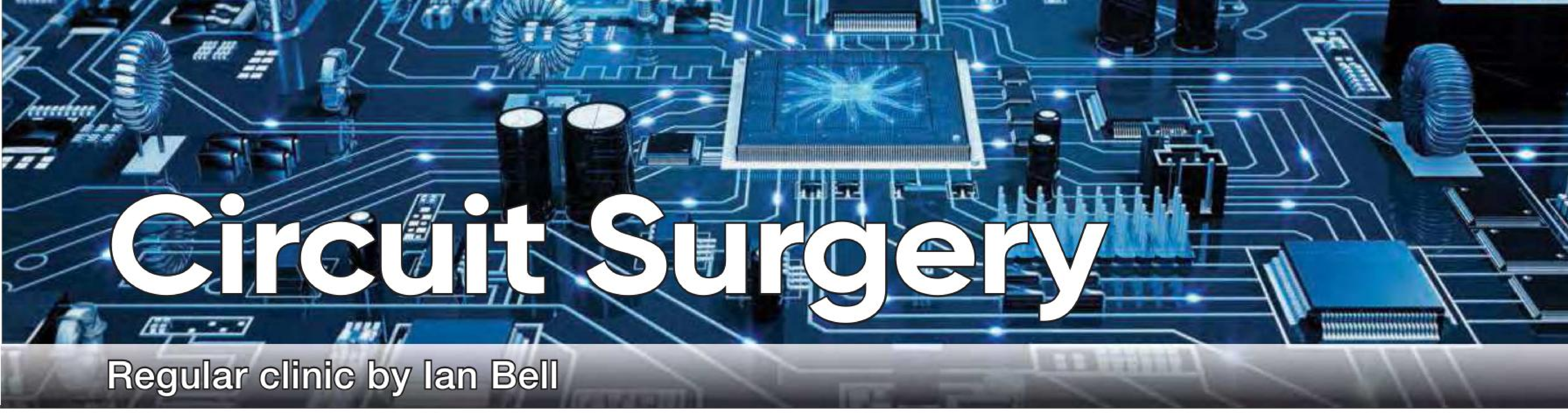
We hope you found this article interesting. If you'd like to try Flowcode for free then just go to **https://flowcode.co.uk/download/** and download the code. You'll get a 30-day free trial of the full version – but that's not all. Even after the 30 days are up your copy of Flowcode will continue to work, but at a reduced level with a limit on the size of program you can run and access to a more basic set of parts. However, for beginners it is still an ideal platform with which you can build and run programs, on for example, the Arduino Uno/RedBoard or a PIC 16F88 that we are using in this project.

Only when you are sure that you want to use Flowcode do you need to buy *inexpensive* access to say a Raspberry Pi or Bluetooth module. (See: **https://flowcode.co.uk/buy/process/** for all the modules available and what they contain.) What's more, as soon as you buy *any* module, the restrictions on the size of your code are removed.

If you get stuck with anything relating to Flowcode use (installation, software, creating flowcharts, compiling to hardware, hardware not working…) then there are forums set up to help you: **https://www.flowcode.co.uk/forums**

PE discount!

One more thing – Flowcode is deliberately designed to be inexpensive, but *PE* readers can get a further 20% discount when they use the code **PE20** at checkout.



Circuit Surgery

Regular clinic by Ian Bell

Distortion and distortion circuits – Part 3

For the past two months we have been looking at distortion – the effect of non-linearities in circuits such as amplifiers on the shape of their output waveforms. Distortion is often an unwanted characteristic that circuit designers make significant efforts to minimise. The amount of unwanted distortion is commonly measured using Total Harmonic Distortion (THD), particularly in audio applications, which, along with the basic concepts of distortion, was covered in the first article (*PE*, June 2022).

Last month, we looked at signal spectra in the context of distortion. A spectrum is a plot of signal strength against frequency, thus showing the frequencies present in a signal. Periodic waveform can be formed by adding together a set of sinewaves of various frequencies and different amplitudes (known as a Fourier series). Therefore, the spectrum plot of a periodic signal shows a set of peaks at specific frequencies.

The spectrum of the output of a linear circuit will not contain frequencies which were not in the input (although the relative amplitude of the original frequencies may change). If distortion occurs due to non-linearities there will be frequencies present in the output which were not in the input. For a sinewave input, the additional output frequencies due to distortion will be at integer multiples of the input frequencies, that is, harmonics of the input. This is the basis of quantifying distortion using THD. LTspice is able to plot signal spectra and calculate THD, hence providing insights into distortion. However, this requires some care and attention in setting up the simulation, and this was a key part of last month's discussion.

Although often unwanted, distortion also has its uses, including in sound processing effects used by musicians – the most well-known example is probably the distortional pedals used by electric guitar players. In fact, this series of articles was inspired by musical effects projects by John Clarke in *PE* over the past year or two. This month, we are going to look at circuits which can be used to deliberately produce distortion for creative purposes.

Simulation files

Most, but not every month, LTSpice is used to support descriptions and analysis in *Circuit Surgery*.
The examples and files are available for download from the *PE* website.

Clipping

Most musical distortion is obtained from circuits which deliberately cause the signal to be clipped – that is, the amplitude of the signal is limited so that the peaks of the waveform are flattened. This is also referred to as saturation. We will recap what we mean by clipping and look at a couple of key variations (hard/soft and symmetrical/asymmetrical) before considering some of the circuits that can be used to achieve these effects.

If we plot the transfer function (input amplitude vs output amplitude) of an ideal amplifier it will be a perfect straight line for all possible amplitudes (the grey line for `v(y1)` on the plots on the left of Fig.1). Real amplifiers have a maximum output amplitude which results in transfer functions more like those shown by the coloured traces for `v(y2)` and `v(y3)` on the left of Fig.1. These two differ in how sharp the transition is between the linear region at low amplitudes and the fully limited region at high amplitudes. A relatively fast transition is referred to as 'hard clipping' (eg, for `y2`). 'Soft clipping' refers to a more gradual transition (eg, for `y3`). The plots in Fig.1 were obtained from a mathematically defined transfer function discussed in the first article. The grey traces on the right side show the undistorted sinewave output.

For a given amplifier, or other circuit which produces clipping, the amount of distortion depends on the input signal amplitude. This is illustrated in Fig.2, which shows the output from the same transfer function as in Fig.1, but with more than twice the input amplitude (1.3V peak in Fig.1, 3V in Fig.2). As amplitude is increased for a sinewave input, the output of clipping circuits will tend towards producing square waves. Softer clipping functions will produce rounder corners on the square wave, but this will also occur if the distorted signal is low-pass filtered.

The transfer function of a distorting circuit does not have to be symmetrical. Non-symmetrical distortion will occur if the transfer function is asymmetrical, or if a signal with a DC offset is applied to a symmetrical transfer function. The latter case is illustrated in Fig.3. This uses the same signal amplitude as in Fig.1 (1.3V peak) but with a +0.65V DC offset on the input sinewave. This means that

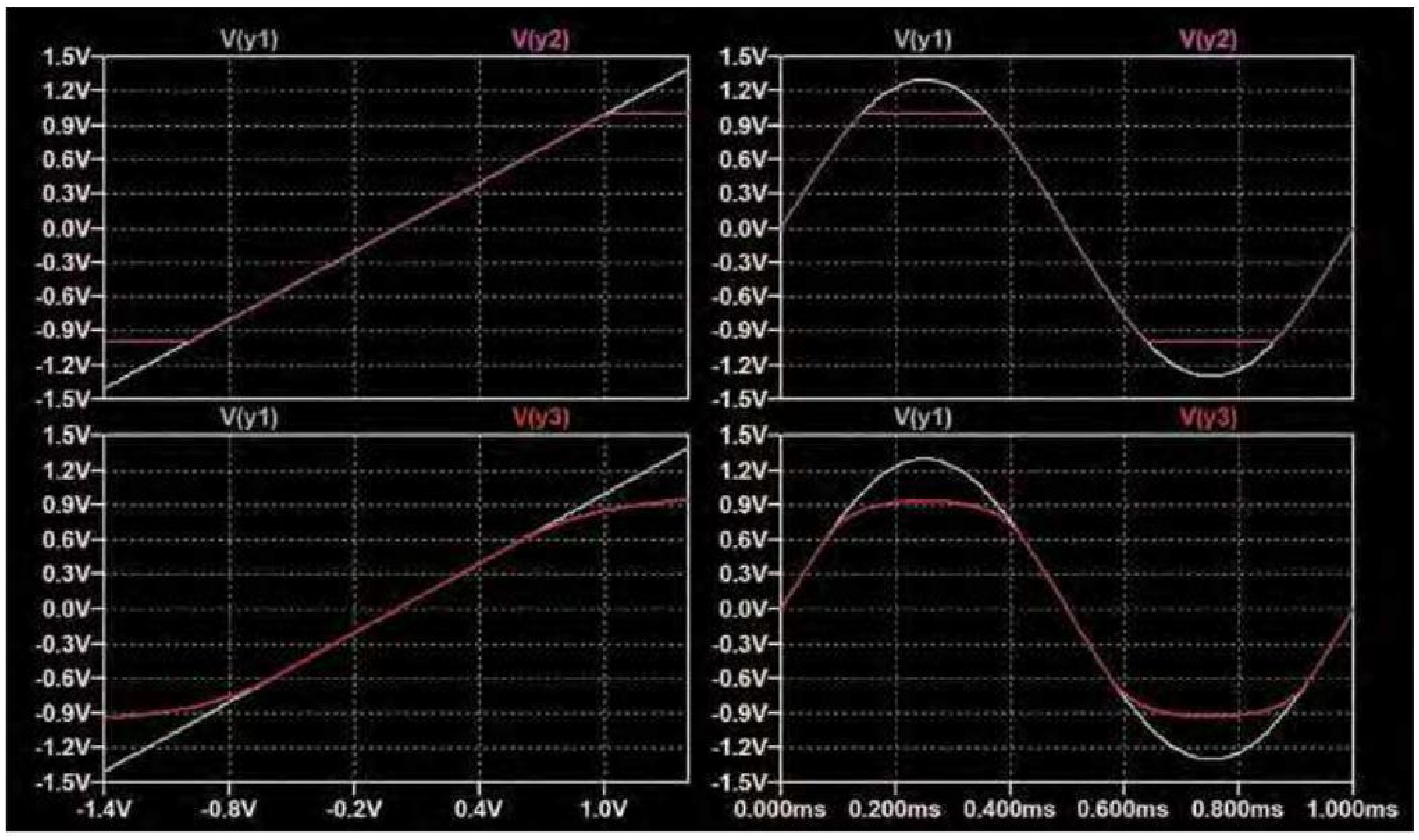


Fig.1. Hard (top) and soft (bottom) clipping.

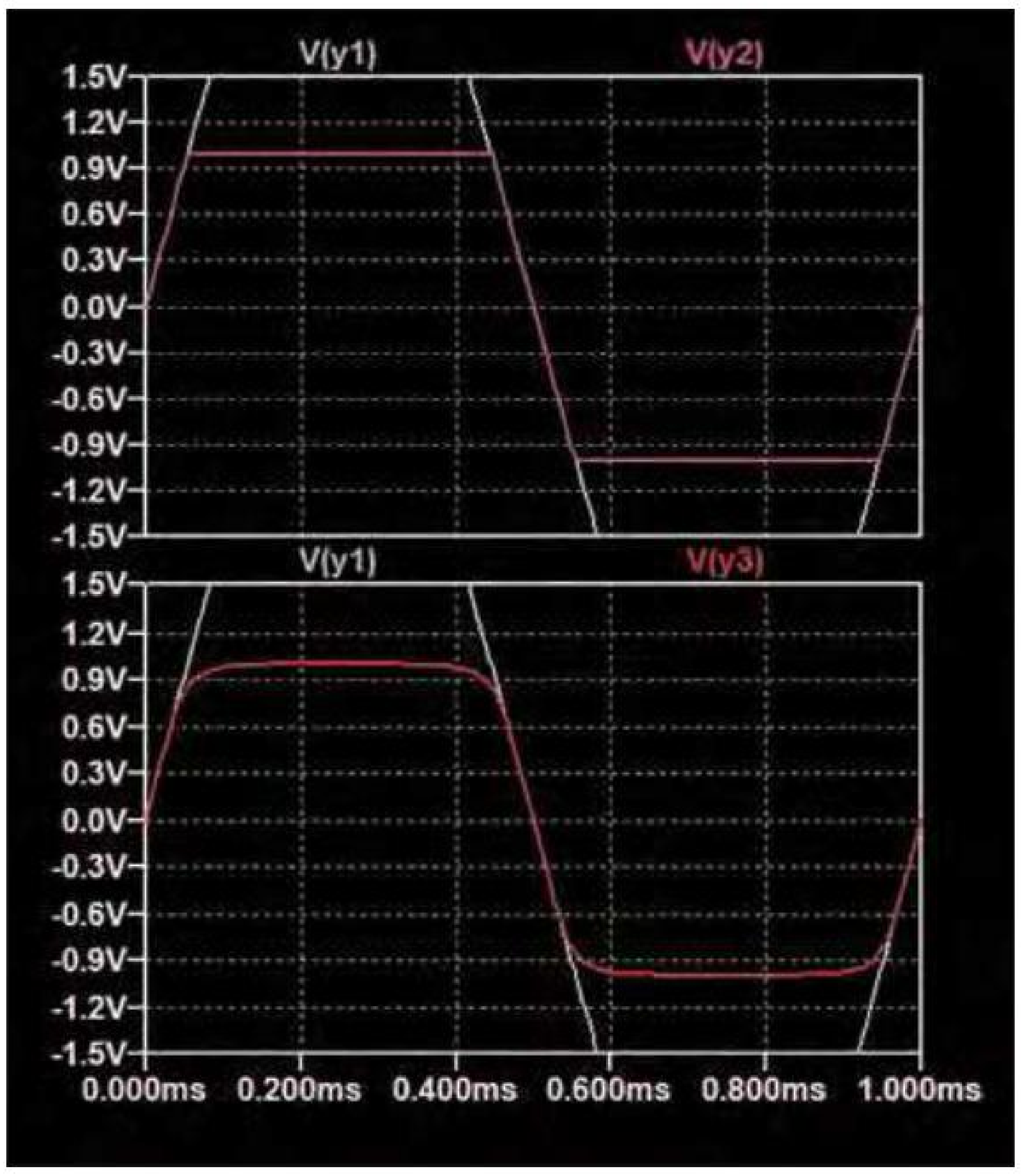


Fig.2. The same clipping function as in Fig.1 with a larger input signal.

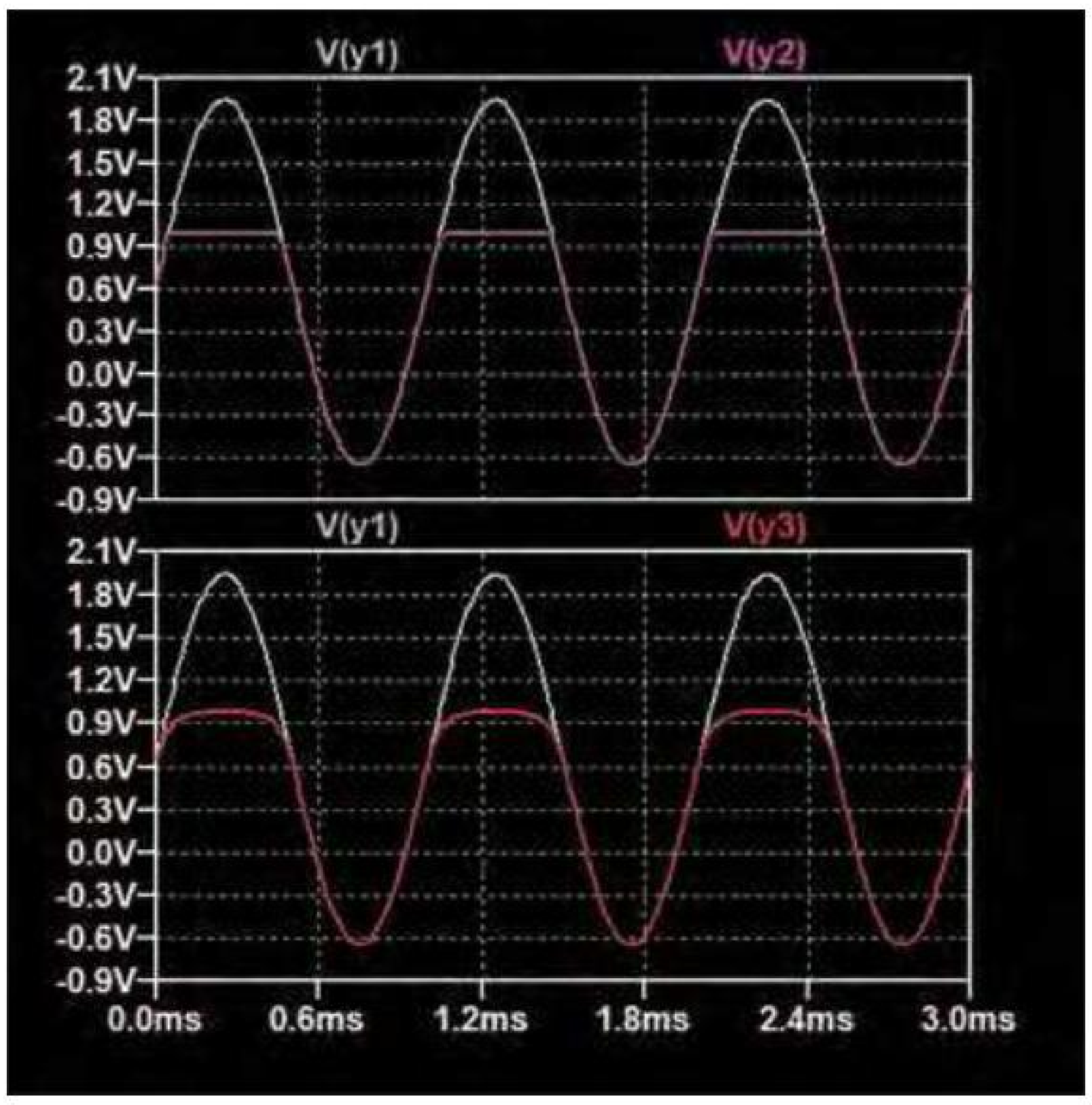


Fig.3. Asymmetrical distortion obtained by applying a sinewave with DC offset to the transfer functions shown on the left-hand side in Fig.1.

the negative peak is at –1.0V and only just reaches the limiting voltage of the clipping function. The negative peak is not significantly distorted, but the positive peak is heavily clipped. The signal in Fig.3 has DC offset, but this is easily removed using a coupling capacitor.

Fig.4 shows an asymmetrical soft clipping transfer function and the resulting output waveform for a 1.3V sinewave with no offset. As discussed in the first article, the soft clipping function used a behavioural source with the function:

```
sgn(v(x))*uplim(abs(v(x)),
lim,0.4)
```

Here, `lim` is the limit or clipping voltage. For asymmetrical clipping, two such versions of the function were used with limit values of 0.5 and 1.0. The two functions were selected using an `if` function:

```
V=if(v(x)+0.5,a,b)
```

Here, *a* and *b* are the two limiting functions. The LTspice function `if(x,y,z)` returns `y` if `x > 0.5`, else `z`. Refer to the first article for the LTspice schematic.

Peddling terminology

Distortion effects pedals used by musicians are often described as providing ‘boost’, ‘overdrive’, ‘distortion’ and ‘fuzz’. The term overdrive was originally used to describe turning up the volume (gain) of a guitar amplifier sufficiently to cause it to clip, thus giving a fuller, grittier sound. In the early days of electric guitars (before the widespread availability of effects pedals), this was necessary to obtain a distortion effect in the amplifier itself. These early amplifiers were built using ‘valve’ – vacuum tube – technology (not transistors/semiconductors), which produced a relatively soft clipping due to the characteristics of the devices. Thus, a pedal marketed as providing overdrive will typically provide soft clipping and aim to emulate the sound obtained by overdriving a valve amplifier.

A ‘boost’ function simply provides gain (amplification) without any clipping. Tone controls may also be provided to control frequency response (provide different gain at different frequencies). As noted above, and seen in Fig.2, increasing signal amplitude produces more distortion, thus a signal boost can be used to help force an amplifier to clip, or increase the amount of distortion provided by the clipping circuit (distortion effects unit) connected to the output of the boost unit.

The terms ‘overdrive’, ‘distortion’ and ‘fuzz’ all refer to signal distortion caused by clipping. As noted above, overdrive typically refers to softer clipping. Effects units marketed as providing distortion and fuzz typically provide harder clipping and/or drive the clipping circuitry at relatively high amplitudes, so are likely to produce outputs which tend more towards square waves, with ‘fuzz’ often implying a more extreme effect than ‘distortion’.

As well as introducing distortion the nonlinear transfer functions of clipping circuits (eg, the left side of Fig.1 and Fig.4) affect the dynamics (volume range) of the output signal. Soft clipping causes smaller signals to be amplified more than larger ones – a process known as compression. For a guitar player (for example) this causes sustain of the notes played. When a string is plucked on an instrument the volume decreases as the vibrations decay. If the gain of the signal increases with decreasing signal this output will tend to remain at a more constant volume than the signal directly from the instrument, resulting in longer lasting notes. This relates to something like the situation in Fig.1, assuming the sinewave shown is around the typical maximum amplitude of a note. In this situation, only the loudest parts of the sound will be distorted. If even relatively quiet notes result in the more extreme clipping like that shown in Fig.2 then there will be relatively little change in output volume for different inputs.

As discussed in depth in the previous article, distortion changes the frequency content of a signal by adding harmonics and (for other than sinewaves) other frequencies. To obtain a good sound from a distortion effect, filtering of the higher additional frequencies using a low-pass filter may be desirable. Also, using frequency-dependent circuitry, distortion can be applied differently at different frequencies. Many distortion units have some form of tone controls (filters) which can adjust the sound, along with the amount of distortion applied.

Diode characteristics

Clipping circuits are essentially amplitude (or voltage) limiters. Diodes are well known for having a near constant voltage across them for a wide range of currents and therefore naturally act as voltage limiters. We can make distortion (clipping) circuits using diodes instead of forcing a full amplifier into saturation. Diode clipping is the basis of many, but not all, distortion effects units.

We can plot the current against voltage (IV) characteristic of diodes using a DC sweep in LTspice – for

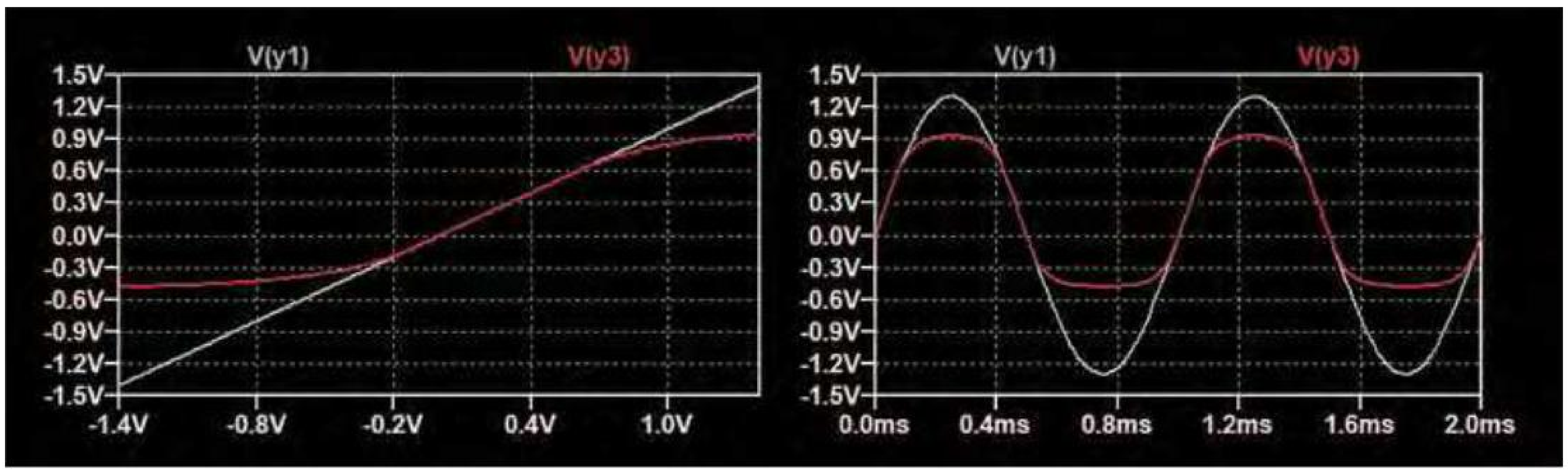
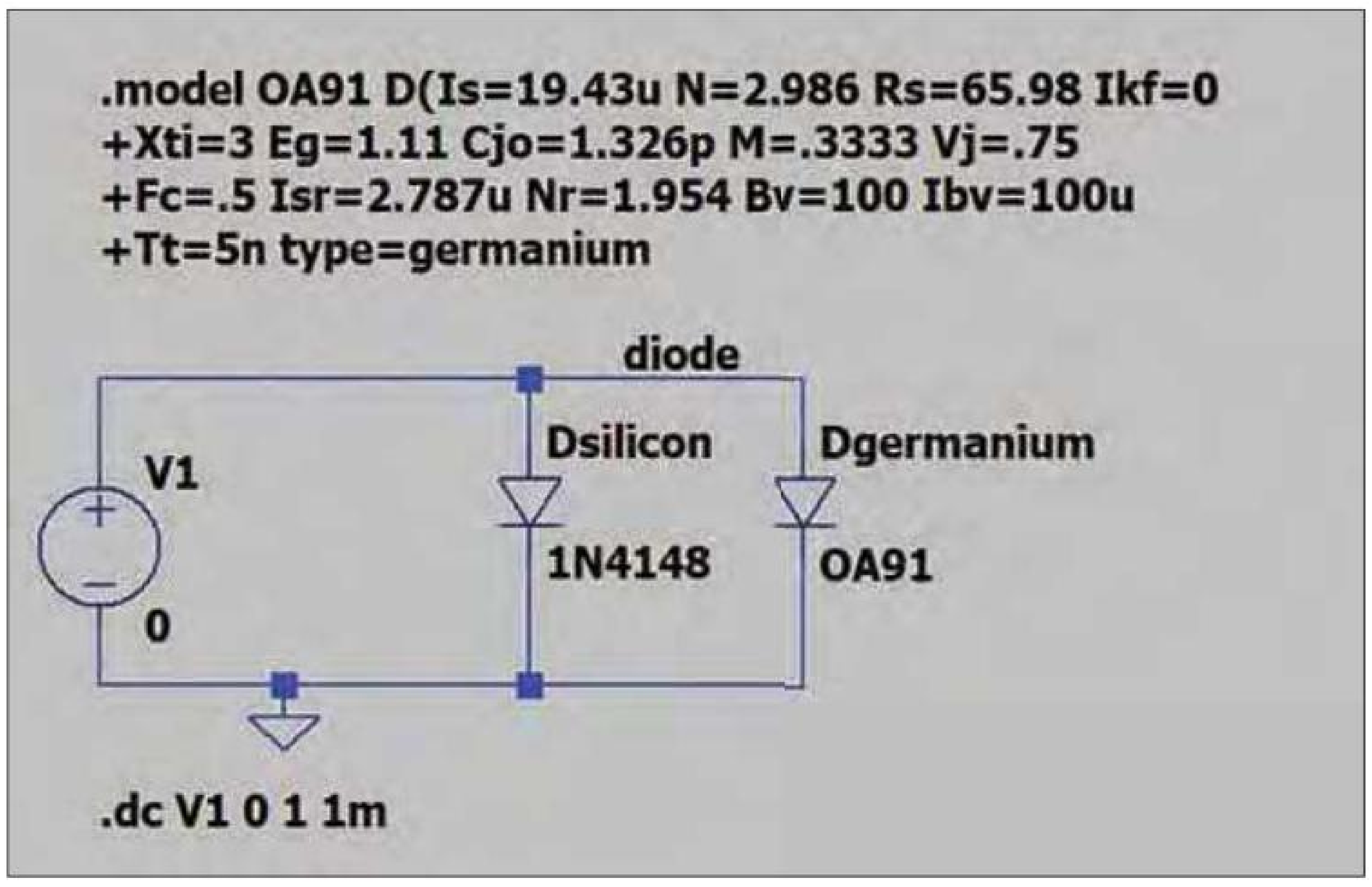


Fig.4. Asymmetrical distortion from an asymmetrical transfer function.



```
.model OA91 D(Is=19.43u N=2.986 Rs=65.98 Ikf=0
+Xti=3 Eg=1.11 Cjo=1.326p M=.3333 Vj=.75
+Fc=.5 Isr=2.787u Nr=1.954 Bv=100 Ibv=100u
+Tt=5n type=germanium
```

```
.dc V1 0 1 1m
```

Fig.5. LTspice circuit to obtain diode IV characteristics.

example, using the setup shown in Fig.5. Here we investigate the IV curves of a very commonly used silicon diode, the 1N4148 and also a germanium diode, the OA91. Germanium diodes were widely used before silicon versions replaced them in most application. Silicon diodes have much higher performance in terms of reverse leakage, maximum reverse voltage, stability, maximum operating temperature and cost. However, germanium diodes still have some niche uses and one of them is in distortion circuits. The OA91 was somewhat arbitrarily selected as an example to use here because the model was found after a quick online search. The current set of diode models provided with LTspice does not include any with `type=germanium`. The `.model` statement is shown in Fig.5.

Fig.6 shows the results from the simulation in Fig.5. There are two key points. First, the germanium diode conducts at a lower voltage than the silicon one – at around 0.2-0.3V rather than 0.6-0.7V. Second, the germanium diode has a less abrupt rise in current as forward voltage increases – this will equate to a softer response when the diode is used in a clipping circuit.

Distortion effects circuits

Fig.7 shows the block diagram of a basic distortion effects unit. This is more of a concept illustration of what happens along the signal path rather than the true schematic structure as the functions may be combined in some implementations. It is also common to have a bypass switch to route the signal past the whole effects circuit to allow the player to switch the effect on and off when needed (not shown). More complex units may have more complex signal paths; for example, to select or mix different distortion effects.

The signal path in Fig.7 starts with a buffer which provides the correct input conditions for the intended source (eg, electric guitar pickup). It may include some filtering, for example to remove very high frequencies such as radio interference. The gain stage is a variable gain amplifier which drives the clipping circuit. As discussed above, and shown in Fig.1 and Fig.2, the signal amplitude into a clipping circuit affects the amount of distortion produced – thus varying the gain varies the amount of distortion for a given input. The clipping stage is a nonlinear circuit with a limiting transfer function (as in Fig.1 and Fig.4). It may be hard or soft, symmetrical, or asymmetrical, or have switches selecting different options, or controls to adjust factors such as asymmetry. The input buffer function may be provided by the same amplifier as the gain stage – it is not necessarily a separate amplifier.

After the clipping stage we have our distorted signal, but its amplitude may need adjusting to be suitable for the destination (amplifier input, or next effects unit in a chain) – this is provided by the output buffer in Fig.7. Some distortion effects also feature tone adjustment – typically a filter to cut high frequencies by a variable amount. In some designs the output stage may not include an active amplifier, and level and tone adjustment may be part of the clipping circuit.

Diode clipping

The circuit shown in Fig.8 is a typical diode clipping circuit used for distortion effects. Not all components are used in all versions and variants exist. The input (typically from a variable gain amplifier, which sets the distortion level) is coupled

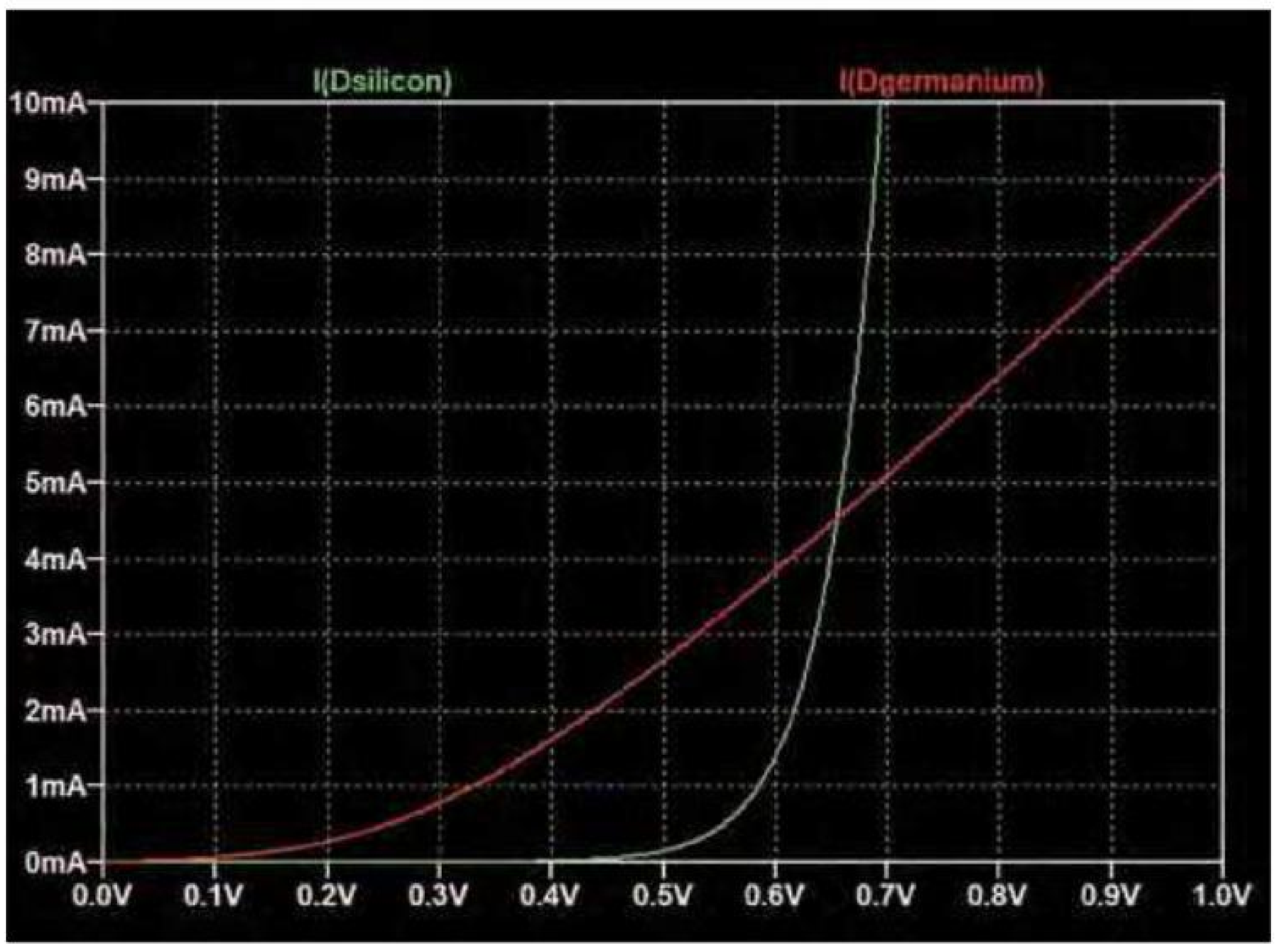


Fig.6. IV characteristics for a silicon (green) and a germanium (red) diode.

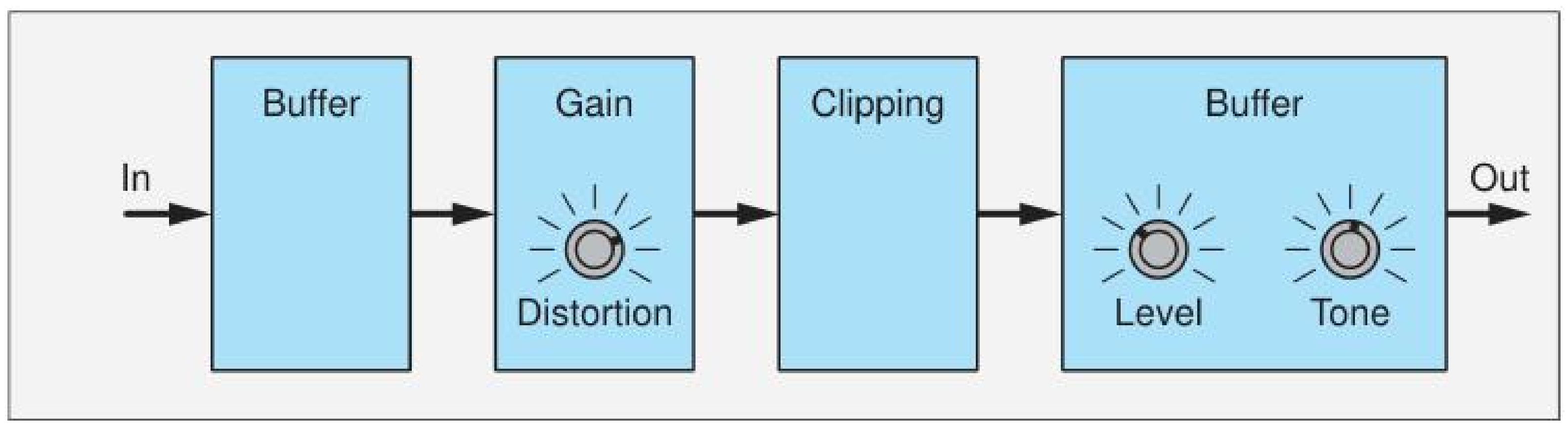
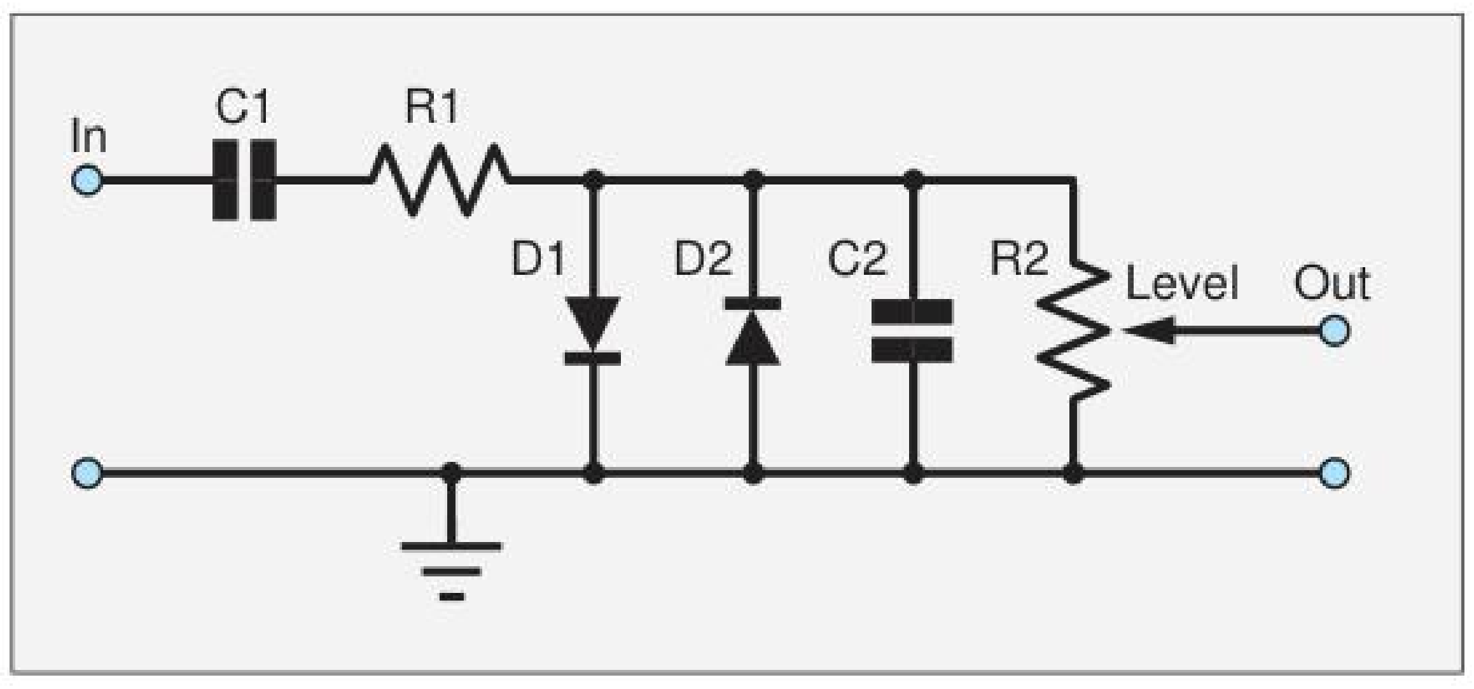
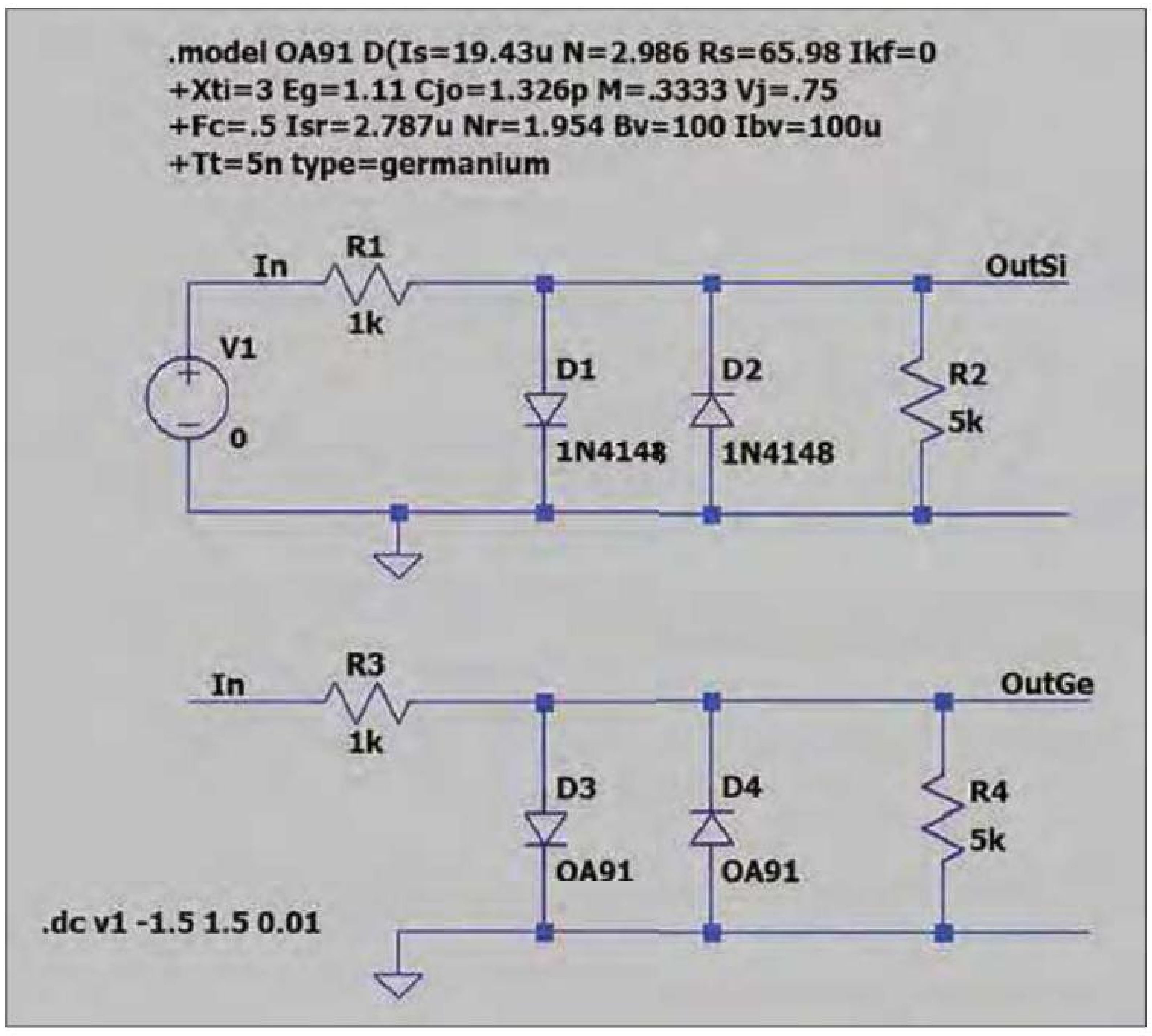


Fig.7. Block diagram of distortion effects unit.

Fig.8. Diode clipping circuit.





```
.model OA91 D(Is=19.43u N=2.986 Rs=65.98 Ikf=0
+Xti=3 Eg=1.11 Cjo=1.326p M=.3333 Vj=.75
+Fc=.5 Isr=2.787u Nr=1.954 Bv=100 Ibv=100u
+Tt=5n type=germanium
```

```
.dc v1 -1.5 1.5 0.01
```

Fig.9. LTspice schematic for simulating the diode clipping circuit with two different diodes.

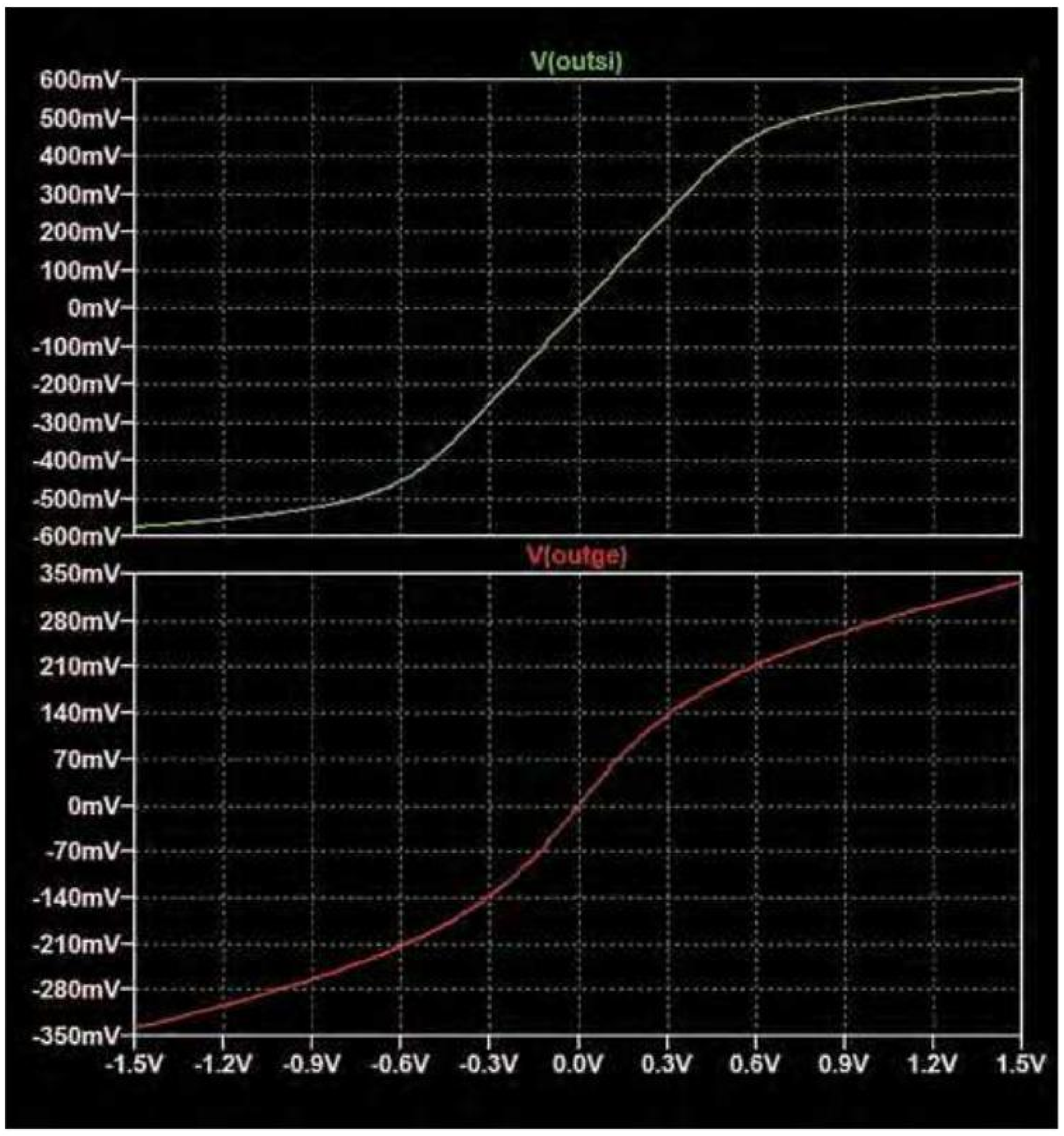


Fig.10. Simulation results from the circuit in Fig.9 – the transfer functions of the clipping circuit with different diodes (top: silicon, bottom: germanium).

via capacitor C1, which removes any DC offset, prevents the diodes from disrupting the bias of the amplifier stage (where applicable), and reduces gain at low frequencies. C1 is typically a few microfarads.

The maximum output signal is limited by (clipped) at the forward voltages of the diodes (D1 and D2). Two diodes are used to cover the positive and negative half cycles of the waveform. Asymmetrical clipping can be achieved by using different types of diode for D1 and D2, or by using a different number of diodes in series in each direction. Diode options are standard silicon didoes, Schottky diodes, germanium diodes and LEDs. Silicon and germanium diodes were compared above. Schottky diodes have a similar characteristic shape to silicon, but a lower forward voltage – closer to germanium at around 0.2V – this is like shifting the silicon curve in Fig.6 to the left. LEDs have forward voltages in the range 1.2 to 4V, depending on the colour.

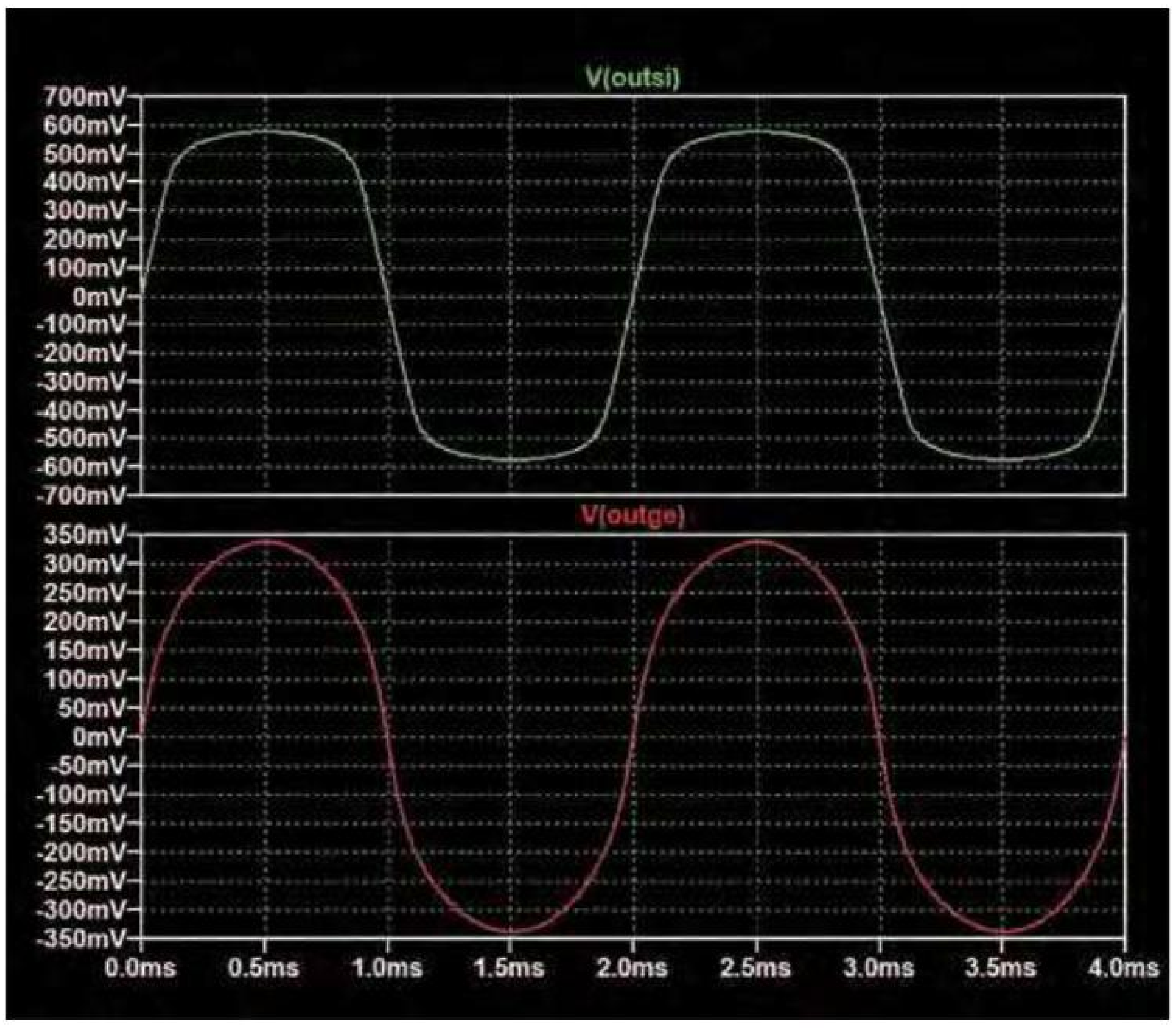


Fig.11. Transient simulation results for a sinewave input to the diode clipping circuit with different diodes (top: silicon, bottom: germanium).

R1 limits the current in the diodes, and R1, R2 and C2 form a low-pass filter which can be used to remove the higher harmonics from the distorted waveform. Potentiometer R2 allows a proportion of the diode signal to be output, providing level adjustment. This can be directly used as the output or passed to an amplifier circuit which may have a tone control. R1 and R2 are typically a few kilohms and C2 is typically a few nanofarads.

Simulation example

The circuit in Fig.9 is an LTspice version of Fig.8 without low-pass capacitor C2. This is so that the full effect of the diodes on the harmonic content can be observed. The input coupling capacitor (C1) is also not included – it is not needed with an ideal signal source, and specifically in Fig.8 we need DC coupling to obtain the transfer function. The results in Fig.10 show that the transfer function is similar to the soft function in Fig.1, with the germanium diode circuit having a softer response.

If we change the voltage source in the circuit in Fig.9 to produce a sinewave and run a transient simulation instead of a DC sweep:

V1 source: `SINE(0 1.5 500)`

Simulation command: `.tran 0 50m 0 10n`

We get the results shown in Fig.11, which shows that the silicon diodes result in harder clipping (or a more square-wave shape) than the germanium diodes. The results from the transient simulation can be used to obtain spectra of the output signals using the LTspice FFT function (taking account of the requirements for doing this effectively, which were discussed in detail last month). The results are shown in Fig.12; using a linear frequency plot here makes it easier to see which harmonic is which. They show that both circuits add odd harmonics, but not even ones. The relative levels of different harmonics are different for the two types of diode. The difference in harmonic content will result in different effects on note timbre – implementing the circuit with different diodes will sound different.

In order to make the spectra easier to compare they have been normalised so that the fundamental (input sine frequency of 500Hz) is at 0dB in both cases. This was done by measuring the fundamental peak on the initial result and calculating the scaling factor required to shift up to 0dB. The scaled signal was then plotted. For example, initially the fundamental peak of the silicon circuit was at –6.34dB. The scaling required is $1/10^{-6.34/20} = 2.075$, so the trace was edited to become `2.075*V(outsi)` – right click the trace title to do this.

Listening test

Given that we are discussing sound processing circuits, it would be useful to be able to listen to the results. Of course, ultimately, we'd want to do listening tests on a real circuit if we were actually developing a distortion effects unit. However, just for fun, or for quickly investigating a range of possible circuits, LTspice can provide listening opportunities via WAV files. For example, you can use a short recoding of (say) a note from a guitar as the input, and listen to the simulated results.

We discussed WAV files in detail in July 2020, so we will be briefer here. For input, change the 'value' of an LTspice source to the form:

```
wavefile=filename
```

Where `filename` is the name of a WAV file in the same folder as the schematic, or the full path to the file if it is elsewhere. By

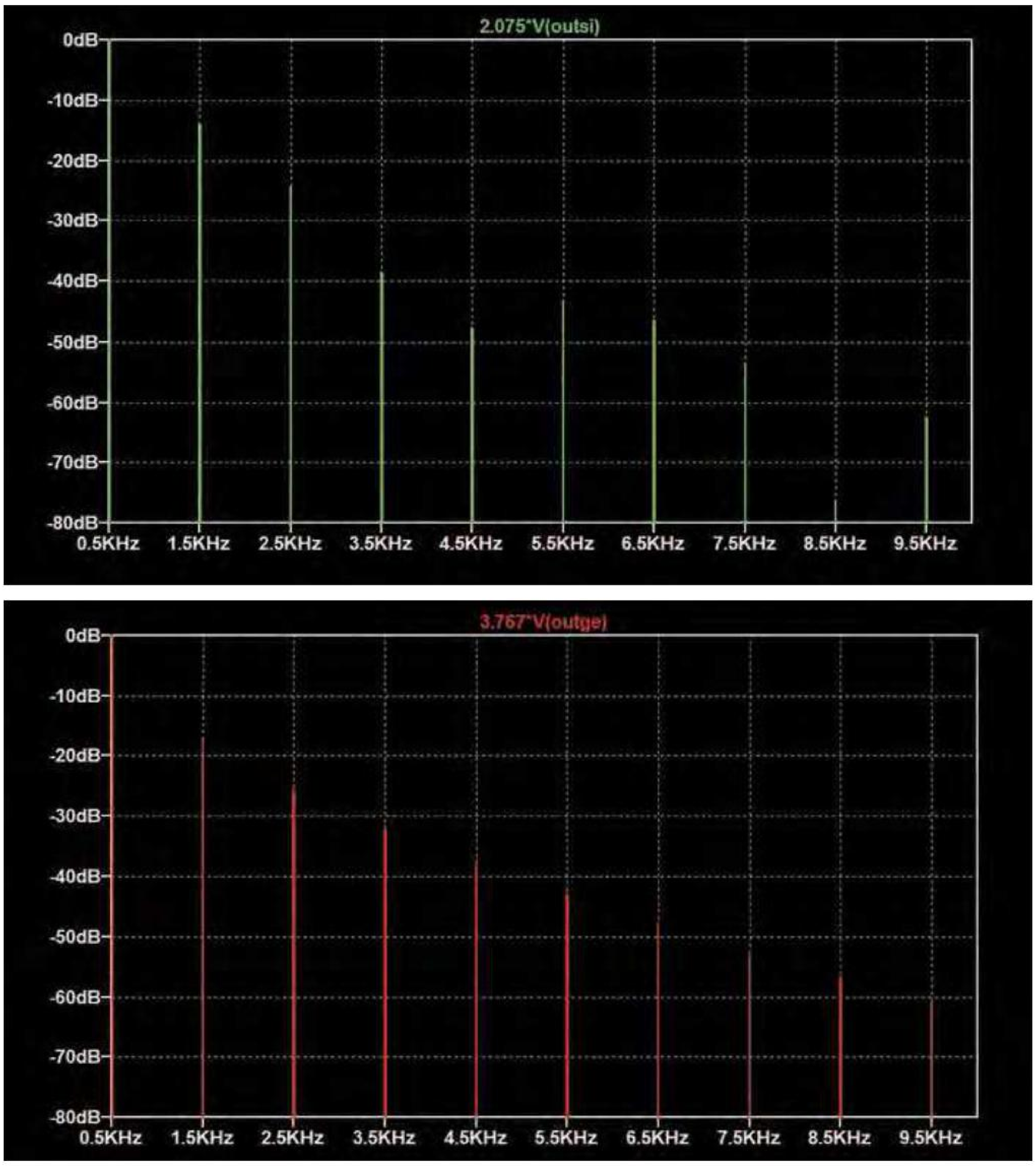


Fig.12. Spectra of the signals from Fig.11 (top: silicon, bottom: germanium).

default, the signal used will be the first channel in the WAV file (typically there are two channels for stereo). To write a WAV file you need to place an LTspice `.wave` directive on the schematic. For audio signals it is best to configure this to a standard format such as stereo 16-bit 44.1 kHz (as used for CDs). For example, to output voltage `out1` to both stereo channels in file `output1.wav` use:

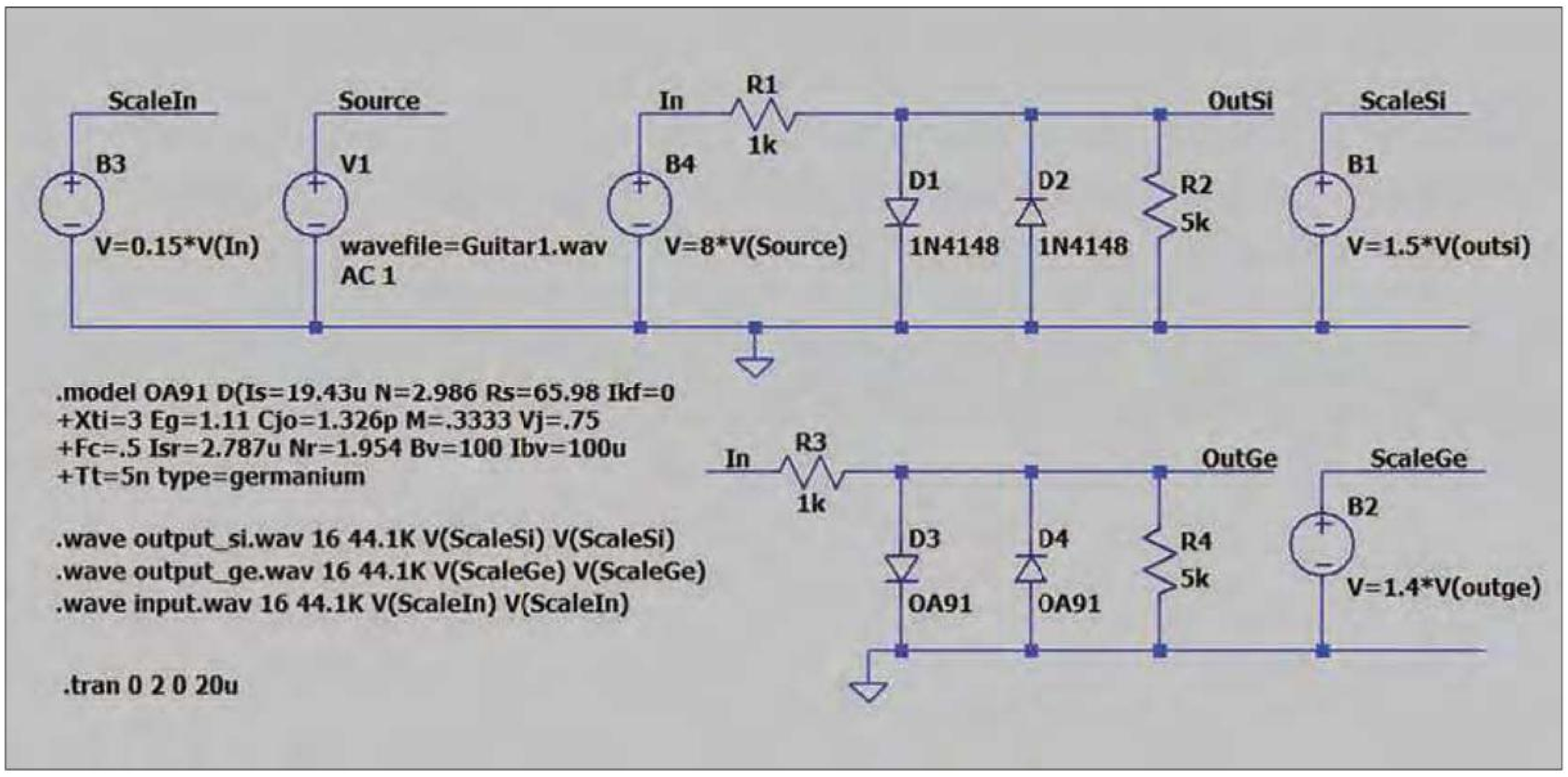
```
.wave output1.wav 16 44.1K
V(out1) V(out1)
```

One key issue is that the WAV file maximum amplitude is 1V. We typically have to scale both the input and outputs to fit with the circuit amplitudes. This is straightforward to do using behavioural sources, but for the output an initial simulation will often be required to measure the peak level so the scaling can be set appropriately. The circuit in Fig.13 is a version of Fig.9 configured for WAV input and output. This was tested using a guitar sample found online (**freewavesamples.com**) – the change in timbre of the output and the difference between the two diode types could be heard.

Other circuits

The diode clipping circuit discussed here is not the only circuit that can be used. Another common approach is to use diodes in the feedback of an op amp amplifier – this is similar to the logarithmic amplifiers discussed in December 2021. A typical configuration is shown in Fig.14, but as before there are variants of this circuit. In this circuit, R1 and R2 set the gain for AC signals as a conventional non-inverting amplifier (gains of 10s to 100s are typical), but C2 blocks DC giving 100% feedback and hence unity gain for DC (not in all variants). The diodes limit the output amplitude to cause clipping. As before, different diodes, or numbers of diodes, can be used for asymmetry. R2 is typically a variable resistor – changing the gain controls the amount of distortion. C1 is typically a small capacitor to reduce gain at high frequencies to reduce the chance of instability. The shape of the transfer function of this circuit is a little different from the diode clipper discussed above, so it will sound different. Some distortion pedals combine both circuits.

The *Nutube Guitar Overdrive and Distortion Pedal* by John Clarke (*PE*, March 2021) uses a different approach – by driving an amplifier into clipping rather than using diodes. The amplifier is a common-cathode stage built using the Nutube low-voltage triode. Asymmetric clipping is achieved by varying the DC bias of the input.



```
.model OA91 D(Is=19.43u N=2.986 Rs=65.98 Ikf=0
+Xti=3 Eg=1.11 Cjo=1.326p M=.3333 Vj=.75
+Fc=.5 Isr=2.787u Nr=1.954 Bv=100 Ibv=100u
+Tt=5n type=germanium

.wave output_si.wav 16 44.1K V(ScaleSi) V(ScaleSi)
.wave output_ge.wav 16 44.1K V(ScaleGe) V(ScaleGe)
.wave input.wav 16 44.1K V(ScaleIn) V(ScaleIn)

.tran 0 2 0 20u
```

Fig.13. LTspice schematic for simulating the diode clipping circuit with WAV file input and output.

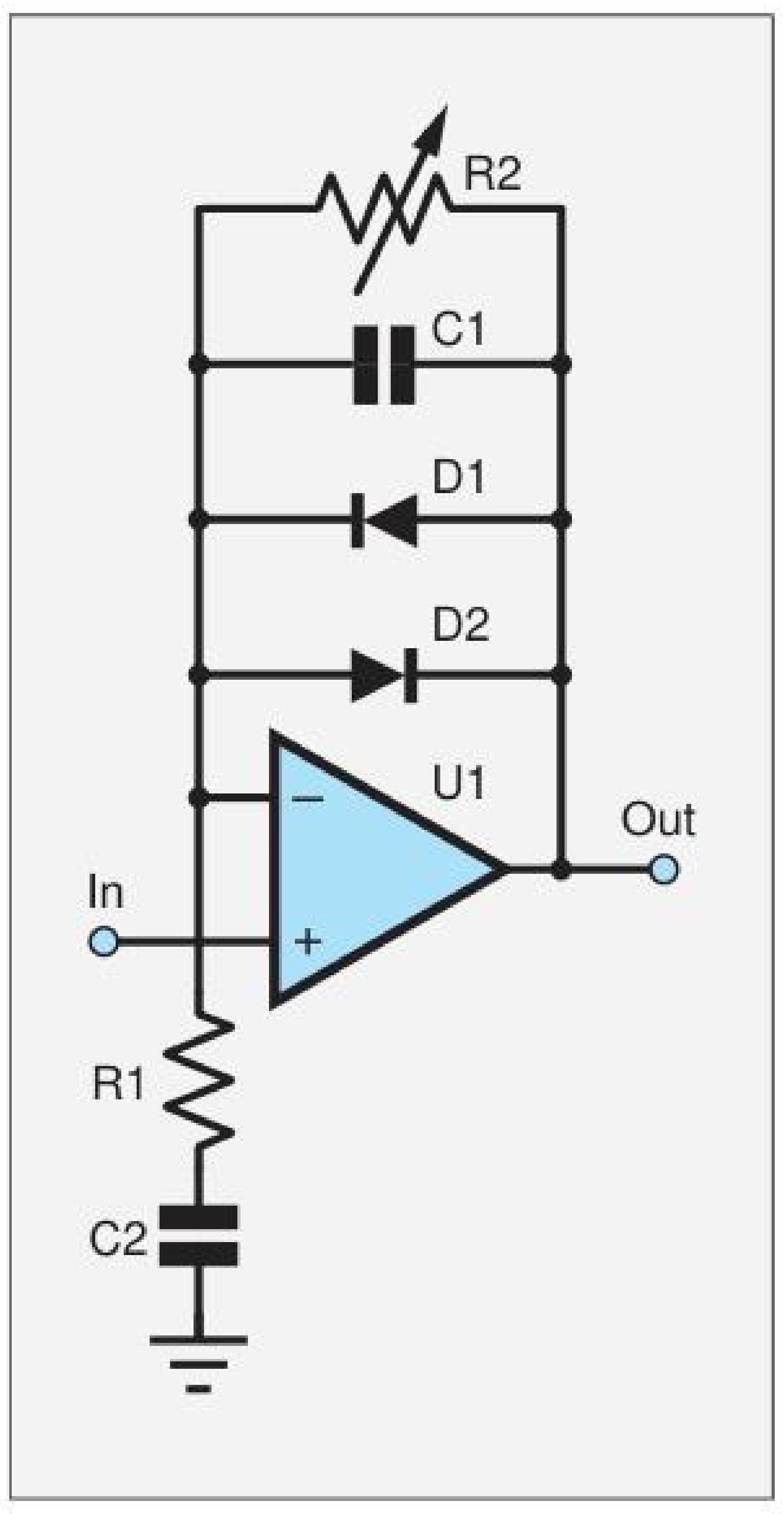


Fig.14. Op amp-based distortion circuit.

Practical Electronics PCB SERVICE

AUGUST 2022

PROJECT	CODE	PRICE
Multi-purpose Battery Manager – I/O Expander module	11104212	5.95
Multi-purpose Battery Manager – Switch Module	11104211	8.95
Simple MIDI Music Keyboard (for 8 switches)	23101213	6.95
Nano Pong	08105212	7.95

JULY 2022

PROJECT	CODE	PRICE
Silicon Labs AM/FM/SW Radio	CSE210301C	10.95
Level Crossing Controller	09108211	6.95

JUNE 2022

PROJECT	CODE	PRICE
Full-wave Motor Speed Controller	1010221	8.95
PIC Programming Helper for 8-pin PICs only	24106211	7.95
PIC Programming Helper for 8, 14 or 20-pin PICs	24106212	10.95
Advanced GPS Computer	05102211	9.95

MAY 2022

PROJECT	CODE	PRICE
Bus board PCB for Analogue Vocoder	AO1-MAY22	10.95
Complete set of 14 PCBs for Analogue Vocoder	AO2-MAY22	97.95
Programmed EEPROM for Digital FX Unit	FX1-MAY22	10.95
Programmed PIC for Digital FX Unit using potentiometer	FX2-MAY22	8.95

APRIL 2022

PROJECT	CODE	PRICE
64-key MIDI Matrix shield	23101211	8.95
64-key MIDI Matrix switch board	23101212	11.95
High-current Battery Balancer	14102211	10.95
Digital FX Unit – using potentiometer	01102211	9.95
Digital FX Unit – using BCD switch	01102212	9.95
Universal Audio PSU	AO1-APR22	11.95

MARCH 2022

PROJECT	CODE	PRICE
Mini Isolated Serial Link	24102211	£5.95
Busy Loo Indicator	16112201	£5.95
Analogue Vocoder – Band-pass filter board	AO1-MAR22	9.95
Analogue Vocoder – HP/LP filter board	AO2-MAR22	9.95

FEBRUARY 2022

PROJECT	CODE	PRICE
Arduino-based Power Supply	18106201	9.95
Battery Monitor Logger	11106201	10.95
Electronic Wind Chimes	23011201	10.95
Analogue Vocoder – Driver Amplifier	AO-FEB22	8.95

JANUARY 2022

PROJECT	CODE	PRICE
Vintage battery Radio Li-ion Power Supply	11111201	9.95
MiniHeart: A Miniature Heartbeat Simulator	01109201	8.95

DECEMBER 2021

PROJECT	CODE	PRICE
AM/FM/SW Digital Receiver	CSE200902A	13.95
Balanced Input and Attenuator for USB CODEC	01106202	11.95

NOVEMBER 2021

PROJECT	CODE	PRICE
Dual Battery Lifesaver	11111202	£6.95

OCTOBER 2021

PROJECT	CODE	PRICE
Mini Wi-Fi LCD BackPack	24106201	£8.95

SEPTEMBER 2021

PROJECT	CODE	PRICE
USB SuperCodec PCB	01106201	£14.95
Audio DDS Oscillator PCB	01110201	£5.95
Audio DDS Oscillator rotary encoder	01110201-ENC	6.95
Programming Adaptor Board for Audio DDS Oscillator	01110202	£5.95
High-power Ultrasonic Cleaner main PCB	04105201	£14.95
High-power Ultrasonic Cleaner front-panel PCB	04105202	
Night Keeper Lighthouse PCB	08110201	£5.95

AUGUST 2021

PROJECT	CODE	PRICE
Ol' Timer PCB	19104201	£11.95
Ol' Timer 8x8 RGB LED module using WS2812B	19104201-88	£8.95
Ol' Timer set of acrylic case pieces and spacer	19104201-ACR	£8.75
Ol' Timer DS3231 RTC IC wide SOIC-16	19104201-RTC	£5.95
Wideband Digital RF Power Meter	04106201	£9.75
Switchmode 78xx regulators ***(PACK of 5!)***	18105201	£7.95
Cool Beans SMAD display	CB-AUG21	£11.95

JULY 2021

PROJECT	CODE	PRICE
ATtiny816 Breakout / Dev Board with Capacitive Touch	24110181	£9.75
IR Remote Control Assistant (Jaycar version)	15005201	£8.95
IR Remote Control Assistant (Altronics version)	15005202	£8.95
PIC18F Development Board	PNM-JUL21	£12.95
Microphone Preamplifier	AO-JUL21	£11.95

JUNE 2021

PROJECT	CODE	PRICE
Roadies' Test Signal Generator (surface-mount version)	01005201	£8.95
Roadies' Test Signal Generator (through-hole version)	01005202	£9.95
Touchscreen Wide-range RCL Box (Resistor module)	04104201	£18.95
Touchscreen Wide-range RCL Box (Ind/Cap module)	04104202	
KickStart Part 3 – Gyrator-based Audio Filter	KS3-2021	£7.95

MAY 2021

PROJECT	CODE	PRICE
7-Band Equaliser (Mono)	01104201	£8.95
7-Band Equaliser (Stereo)	01104202	£10.95
Car Altimeter	05105201	£7.95

APRIL 2021

PROJECT	CODE	PRICE
Reflow Oven – DSP Active Crossover (CPU)	01106193	£19.95
Reflow Oven – DSP Active Crossover (Front panel)	01106195	
Reflow Oven – DSP Active Crossover (LCD)	01106196	
Frequency Reference Signal Distributor	CSE200103	£8.95

MARCH 2021

PROJECT	CODE	PRICE
Nutube Guitar Effects Pedal	01102201	£12.95
Programmable Thermal Regulator (Peltier Interface)	21109181	£18.95
Programmable Thermal Regulator (Peltier Driver)	21109182	
Tunable HF Preamp	CSE190502	£8.95

FEBRUARY 2021

PROJECT	CODE	PRICE
4G Remote Monitoring	27111191	£9.95

JANUARY 2021

PROJECT	CODE	PRICE
Nutube Valve Preamplifier	01112191	£12.95
Arduino DCC Controller	09207181	£10.95

DECEMBER 2020

PROJECT	CODE	PRICE
Pseudo-Random Sequence Generator	16106191	£7.95
Clever Charger	14107191	£11.95
PE Theremin Amplifier	AO-1220-01	£8.95

NOVEMBER 2020

PROJECT	CODE	PRICE
LED Christmas Tree (1 off)	16107181-1	£6.95
LED Christmas Tree (4 off)	16107181-2	£14.95
LED Christmas Tree (12 off)	16107181-3	£24.95
LED Christmas Tree (20 off)	16107181-4	£34.95
USB/SPI Interface Board	16107182	£5.95
45V/8A Power Supply PCB *plus* acrylic spacer	18111181	£14.95
45V/8A Power Supply front panel five-way display bezel	18111181-BZ	£3.95
Five-way LCD Panel Meter/Display	18111182	£7.95

OCTOBER 2020

PROJECT	CODE	PRICE
Digital Audio Millivoltmeter	04108191	£9.95
Precision Signal Amplifier	04107191	£7.95

PCBs for most recent *PE/EPE* constructional projects are available. From the July 2013 issue onwards, PCBs with eight-digit codes have silk screen overlays and, where applicable, are double-sided, have plated-through holes, and solder mask. They are similar to photos in the project articles. Earlier PCBs are likely to be more basic and may not include silk screen overlay, be single-sided, lack plated-through holes and solder mask.

Always check price and availability in the latest issue or online. A large number of older boards are listed for ordering on our website.

In most cases we do not supply kits or components for our projects. For older projects it is important to check the availability of all components before purchasing PCBs.

Back issues of articles are available – see Back Issues page for details.

CLASSIFIED ADVERTISING

Practical Electronics

If you want your advertisements to be seen by the largest readership at the most economical price then our classified page offers excellent value. The rate for semi-display space is £10 (+VAT) per centimetre high, with a minimum height of 2·5cm. All semi-display adverts have a width of 5.5cm. The prepaid rate for classified adverts is 40p (+VAT) per word (minimum 12 words).

Cheques are made payable to 'Practical Electronics'. **VAT must be added**. Advertisements with remittance should be sent to: Practical Electronics, 113 Lynwood Drive, Wimborne, Dorset, BH21 1UU.

Tel 07973518682 Email: pe@electronpublishing.com

For rates and further information on display and classified advertising please contact our Advertisement Manager, Matt Pulzer – see below.

Practical Electronics reaches more UK readers than any other UK monthly hobby electronics magazine. Our sales figures prove it.

We have been the leading monthly magazine in this market for the last twenty-seven years.

BOWOOD ELECTRONICS LTD

Suppliers of Electronic Components

www.bowood-electronics.co.uk

Unit 10, Boythorpe Business Park, Dock Walk, Chesterfield,
Derbyshire S40 2QR. Sales: 01246 200 222

Send large letter stamp for Catalogue

MISCELLANEOUS

VALVES AND ALLIED COMPONENTS? For free stock list and/or advice, please contact me: geoffdavies337@gmail.com Telephone: 01788 574774

PIC DEVELOPMENT KITS, DTMF kits and modules, CTCSS Encoder and Decoder/Display kits. Visit www.cstech.co.uk

Electrical Industries Charity (EIC)
We help people working in the electrical, electronics and energy community as well as their family members and retirees.

We use workplace programmes that give the industry access to financial grants and a comprehensive range of free and confidential services.
www.electricalcharity.org

electrical industries charity

COAST ELECTRONICS

BREAKOUTS-COMPONENTS-
CONTRACT DESIGN-3D PRINTER PARTS-
MUSICAL-MICROCONTROLLERS
WWW.COASTELECTRONICS.CO.UK

Andrew Kenny – Qualified Patent Agent
EPO UKIPO USPTO
Circuits Electric Machinery Mechatronics
Web: www.akennypatentm.com
Email: Enquiries@akennypatentm.com
Tel: 0789 606 9725

ADVERTISING INDEX

CRICKLEWOOD ELECTRONICS	53
ESR ELECTRONIC COMPONENTS	25
FLOWCODE	6
HAMMOND ELECTRONICS Ltd	13
JPG ELECTRONICS	72
MICROCHIP	Cover (ii)
PEAK ELECTRONIC DESIGN	Cover (iv)
POLABS D.O.O.	29
QUASAR ELECTRONICS	3
SILICON CHIP	5
STEWART OF READING	39
TAG-CONNECT	25

Advertisement offices

Matt Pulzer

Electron Publishing Ltd
1 Buckingham Road
Brighton
East Sussex BN1 3RA

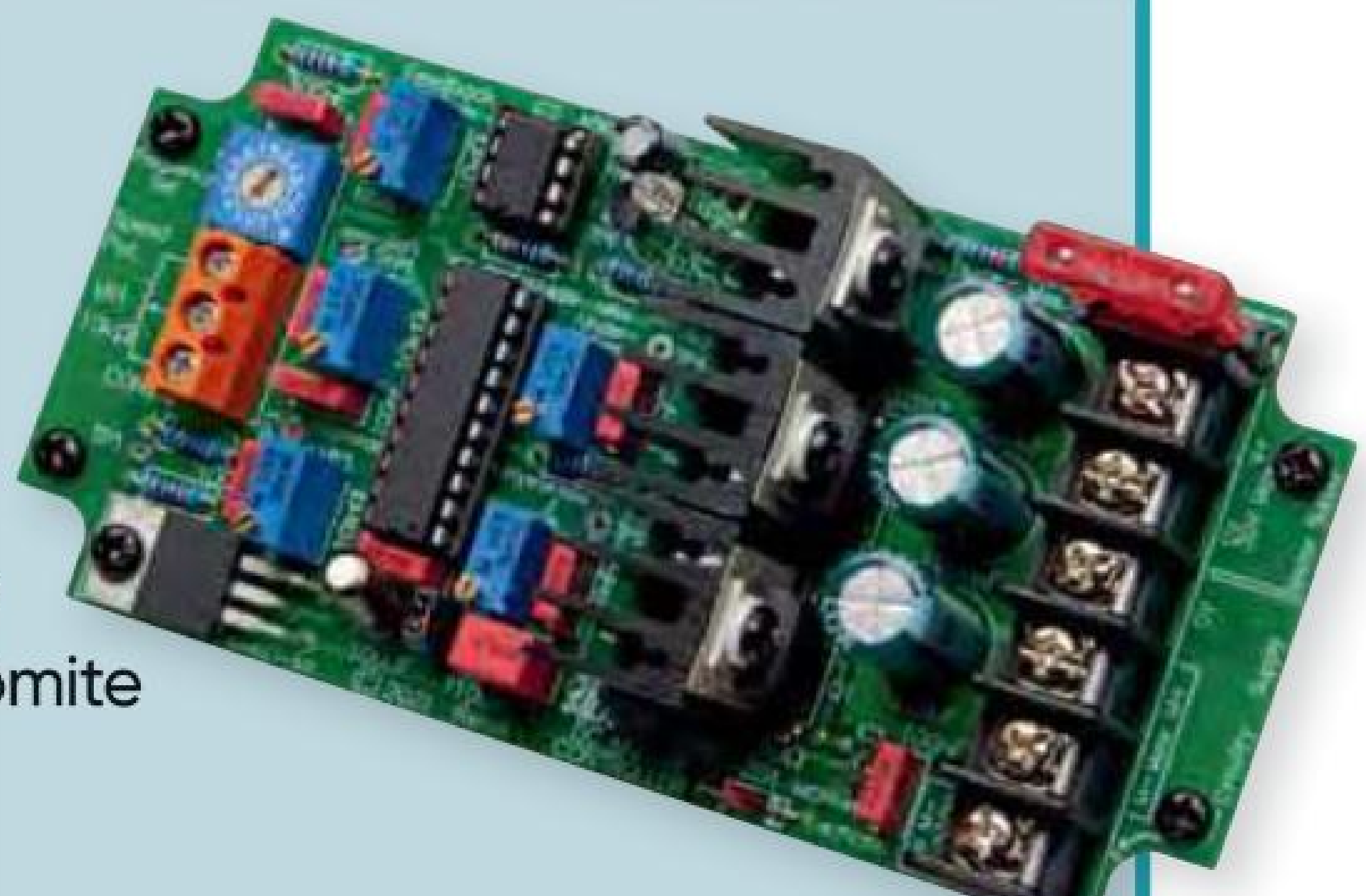
Tel 07973 518682

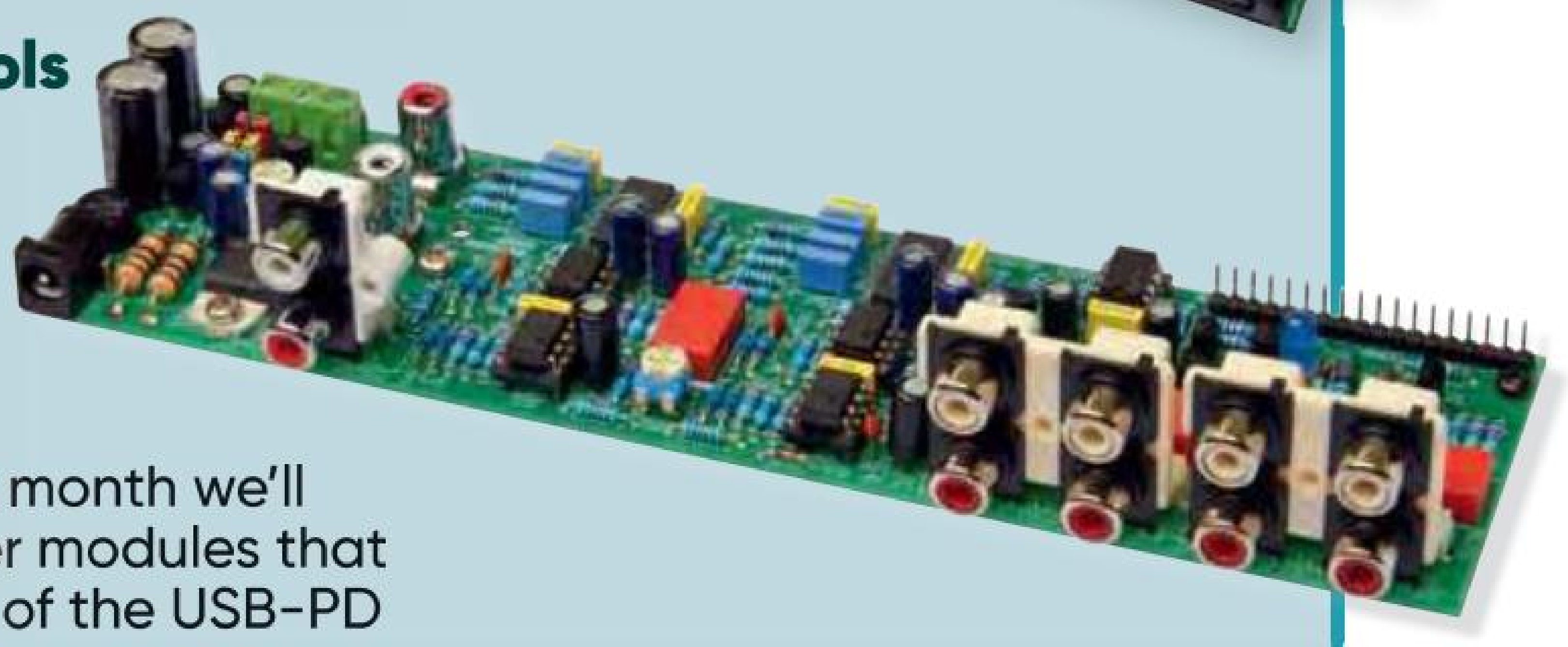
Email pe@electronpublishing.com

Web www.electronpublishing.com

For editorial contact details see page 7.

Next Month – in the September issue





20A DC Motor Speed Controller

This small but powerful speed controller has a 20A rating and is packed with features. It suits a wide range of applications and is simple to build and use. Features include low-battery protection, soft-start and adjustable pulse frequency. It can handle DC motors that run from near 0V up to 30V.

Micromite to Smartphone Connector via Bluetooth

This project demonstrates how to use a Micromite as the heart of an IoT (Internet of Things) device. But there are many other reasons you might wish to connect a Micromite to your Android smartphone, such as making it easy to monitor what your device is doing without going to the trouble or expense of fitting it with an LCD screen.

Touchscreen & Remote Digital Preamp with Tone Controls

This preamp has the best of both worlds: the benefits of digital control such as an intuitive touchscreen interface, presets and remote control, along with the low noise and distortion of an analogue design.

USB-PD Triggers, Decoys & Testers

Following on from the article on USB-PD charging modules, next month we'll examine some of the many low-cost PD trigger/decoy and tester modules that have appeared recently. These allow you to take full advantage of the USB-PD chargers' capabilities and use them as versatile and efficient power supplies.

PLUS!

All your favourite regular columns from *Audio Out, Cool Beans and Circuit Surgery*, to *Make it with Micromite, Techno Talk* and *Net Work*.

On sale 4 August 2022

Content may be subject to change

WELCOME TO JPG ELECTRONICS

Selling Electronics in Chesterfield for 29 Years

Open Monday to Friday 9am to 5:30pm
And Saturday 9:30am to 5pm

- Aerials, Satellite Dishes & LCD Brackets
- Audio Adaptors, Connectors & Leads
- BT, Broadband, Network & USB Leads
- Computer Memory, Hard Drives & Parts
- DJ Equipment, Lighting & Supplies
- Extensive Electronic Components
  - *ICs, Project Boxes, Relays & Resistors*
- Raspberry Pi & Arduino Products
- Replacement Laptop Power Supplies
- Batteries, Fuses, Glue, Tools & Lots more...

T: 01246 211 202
E: sales@jpgelectronics.com
JPG Electronics, Shaw's Row,
Old Road, Chesterfield, S40 2RB
W: www.jpgelectronics.com

JPG Electronics

Retail & Trade Welcome • Free Parking • Google St View Tour: S40 2RB

NEW subscriptions hotline!

Practical Electronics

We have changed the way we sell and renew subscriptions. We now use 'Select Publisher Services' for all print subscriptions – to start a new subscription or renew an existing one you have three choices:

1. Call our NEW print subscription hotline: 01202 087631, or email: pesubs@selectps.com
2. Visit our shop at: www.electronpublishing.com
3. Send a cheque (payable to: 'Practical Electronics') with your details to:
   Practical Electronics Subscriptions, PO Box 6337, Bournemouth BH1 9EH, United Kingdom

Remember, we print the date of the last issue of your current subscription in a box on the address sheet that comes with your copy.

Digital subscribers, please call 01202 880299 or visit: www.electronpublishing.com

Published on approximately the first Thursday of each month by Electron Publishing Limited, 1 Buckingham Road, Brighton, East Sussex BN1 3RA. Printed in England by Acorn Web Offset Ltd., Normanton WF6 1TW. Distributed by Seymour, 86 Newman St., London W1T 3EX. Subscriptions UK: £29.99 (6 months); £54.99 (12 months); £104.99 (2 years). EUROPE: airmail service, £33.99 (6 months); £63.99 (12 months); £119.99 (2 years). REST OF THE WORLD: airmail service, £41.99 (6 months); £77.99 (12 months); £149.99 (2 years). Payments payable to 'Practical Electronics', Practical Electronics Subscriptions, PO Box 6337, Bournemouth BH1 9EH, United Kingdom. Email: pesubs@selectps.com. PRACTICAL ELECTRONICS is sold subject to the following conditions, namely that it shall not, without the written consent of the Publishers first having been given, be lent, resold, hired out or otherwise disposed of by way of Trade at more than the recommended selling price shown on the cover, and that it shall not be lent, resold, hired out or otherwise disposed of in a mutilated condition or in any unauthorised cover by way of Trade or affixed to or as part of any publication or advertising, literary or pictorial matter whatsoever.

Did you know our online shop now sells the current issue of *PE* for £5.49 inc. p&p?

Practical Electronics

The UK's premier electronics and computing maker magazine

Circuit Surgery
Using distortion and distortion circuits

KickStart
Exploring DACs and microcontrollers

Make it with Micromite
MMBASIC + RPi Pico + display = PicoMite Backpack!

WIN! Microchip SAM V71 Xplained Ultra Evaluation Kit WIN!

Multi-purpose Battery Manager

Retro gaming with Nano Pong!

Flowcode Digital Clock Design

Simple MIDI Music Keyboard

PLUS!
Techno Talk – Time for a total rethink?
Cool Beans – Touch-sensitive robots and using servos
Net Work – The irresistible rise of automotive electronics

Aug 2022 £5.49

www.electronpublishing.com | @practicalelec | practicalelectronics

Practical Electronics

and computing maker magazine

Make it with Micromite
Installing MMBASIC on a Raspberry Pi Pico

WIN! Microchip SAM C21 Xplained Pro Evaluation Kit WIN!

Controlling a linear actuator

GPS Computer: on and use

mobile payments

competition

Jul 2022 £5.49

electronpublishing.com | @practicalelec | practicalelectronics

You read that right! We now sell the current issue of your favourite electronics magazine for exactly the same price as in the High Street, but we deliver it straight to your door – and for UK addresses we pay the postage. No need to journey into town to queue outside the newsagent. Just go to our website, set up an account in 30 seconds, order your magazine and we'll do the rest.

www.electronpublishing.com

PEAK®
electronic design ltd

15% discount!

Use code *ESRPE* during checkout
offer expires 31st July 2022!

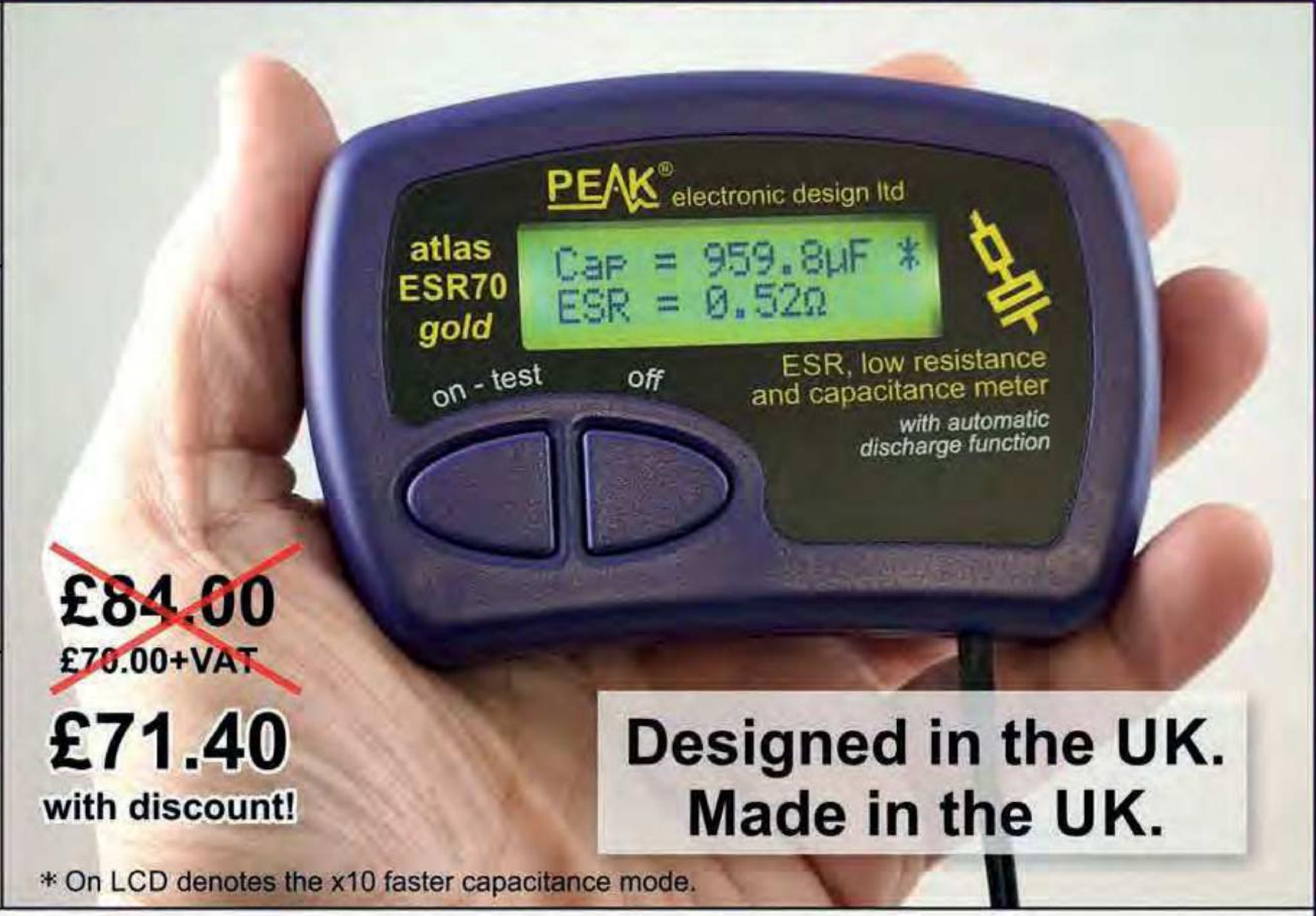
NEW! Peak Atlas ESR70 Gold

Equivalent Series Resistance and Capacitance Meter

Flexible test leads included

16 page
Fully illustrated User Guide

Optional probe types


PEAK® electronic design ltd
atlas
ESR70
gold
Cap = 959.8µF *
ESR = 0.52Ω
on - test
off
ESR, low resistance and capacitance meter
with automatic discharge function

~~£84.00~~
~~£70.00+VAT~~

£71.40
with discount!

Designed in the UK.
Made in the UK.

* On LCD denotes the x10 faster capacitance mode.

What does the new ESR70 Gold do?

The new improved **ESR70 Gold** is great for identifying capacitors (especially electrolytics) that may be suffering from high ESR.

ESR is a great indicator of capacitor condition, especially in applications demanding high ripple currents such as reservoir capacitors, SMPSUs, motor capacitors etc.

The **ESR70 Gold** will even perform a controlled discharge of your caps!

ESR Analysis starts automatically when you apply the probes to the capacitor, even in-circuit.

For out-of-circuit testing, the **ESR70 Gold** also measures capacitance more rapidly than ever.

This **ESR70 Gold** includes removable gold-plated croc-clip probes, long ultra-flexible silicone test leads, alkaline AAA battery and a very comprehensive user guide.

All Peak products carry a 24 month warranty. Even after that period, we offer very low cost repair, update and calibration services.

Gold Features

- Much wider capacitance range 0.3µF to 90,000µF (was 1µF to 22,000µF).
- 10x faster capacitance measurement for large capacitors.
- Improved in-circuit accuracy.
- Improved user options system.
- New triple-slope measurement system vastly reduces the influence of parallel resistance and leakage current on capacitance readings.

Summary Specs

- ESR measurement from 0.00Ω to 40.0Ω. Can be used for simple low-ohms testing too.
- Audible alerts for different ESR levels.
- Automatic controlled discharge before analysis.
- Automatic and manual power-off.
- New backlit display.
- Very long battery life from a single AAA cell.

If you already have an ESR70 that accepts an AAA battery, contact us about having it upgraded to the new Gold spec.

It's only possible to show summary specifications here. Please ask if you'd like detailed data. Further information is also available on our website. Product price refunded if you're not happy.

www.peakelec.co.uk
sales@peakelec.co.uk

Atlas House, 2 Kiln Lane, Harpur Hill Business Park
Buxton, Derbyshire, SK17 9JL, UK. Tel: 01298 70012